An Alternating Trust Region Algorithm for Distributed Linearly Constrained Nonlinear Programs

Application to the AC Optimal Power Flow

Jean-Hubert Hours and Colin N. Jones

Received: date / Accepted: date

Abstract A novel trust region method for solving linearly constrained nonlinear programs is presented. The proposed technique is amenable to a distributed implementation, as its salient ingredient is an alternating projected gradient sweep in place of the Cauchy point computation. It is proven that the algorithm yields a sequence that globally converges to a critical point. As a result of some changes to the standard trust region method, namely a proximal regularisation of the trust region subproblem, it is shown that the local convergence rate is linear with an arbitrarily small ratio. Thus, convergence is locally almost superlinear, under standard regularity assumptions. The proposed method is successfully applied to compute local solutions to alternating current optimal power flow problems in transmission and distribution networks. Moreover, the new mechanism for computing a Cauchy point compares favourably against the standard projected search as for its activity detection properties.

Keywords Nonconvex optimisation, Distributed optimisation, Coordinate gradient descent, Trust region methods

Mathematics Subject Classification 49M27 · 49M37 · 65K05 · 65K10 · 90C06 · 90C26 · 90C30

The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP/2007-2014)/ ERC Grant Agreement n.307608.

1 Introduction

Minimising a separable smooth nonconvex function subject to partially separable coupling equality constraints and separable constraints, appears in many engineering problems such as Distributed Nonlinear Model Predictive Control (DNMPC) (Necoara, I. and Savorgnan, C. and Tran Dinh, Q. and Suykens, J. and Diehl, M., 2009), power systems (Kim, B.H. and Baldick, R., 1997) and wireless networking (Chiang et al, 2007). For such problems involving a large number of agents, which result in large-scale nonconvex Nonlinear Programs (NLP), it may be desirable to perform computations

Correspondence to: Jean-Hubert Hours
Automatic Control Laboratory, Ecole Polytechnique Fédérale de Lausanne, E-mail: jean-hubert.hours@epfl.ch

Colin N. Jones
Automatic Control Laboratory, Ecole Polytechnique Fédérale de Lausanne

in a distributed manner, meaning that all operations are not carried out on one single node, but on multiple nodes spread over a network and that information is exchanged during the optimisation process. Such a strategy may prove useful to reduce the computational burden in the case of extremely large-scale problems. Moreover, autonomy of the agents may be hampered by a purely centralised algorithm. Case in points are cooperative tracking using DNMPC (Hours, J.-H. and Jones, C.N., 2016) or the Optimal Power Flow problem (OPF) over a distribution network (Gan, L. and Li, N. and Topcu, U. and Low, S.H., 2014), into which generating entities may be plugged or unplugged. Moreover, it has been shown in a number of studies that distributing and parallelising computations can lead to significant speed-up in solving large-scale NLPs (Zavala, V.M. and Laird, C.D. and Biegler, L.T., 2008). Splitting operations can be done on distributed memory parallel environments such as clusters (Zavala, V.M. and Laird, C.D. and Biegler, L.T., 2008), or on parallel computing architectures such as Graphical Processing Units (GPU) (Fei, Y. and Guodong, R. and Wang, B. and Wang, W., 2014).

Our objective is to develop nonlinear programming methods in which most of the computations can be distributed or parallelised. Some of the key features of a distributed optimisation strategy are the following:

(i) *Shared memory.* Vectors and matrices involved in the optimisation process are stored on different nodes. This requirement rules out direct linear algebra methods, which require the assembly of matrices on a central unit.
(ii) *Concurrency.* A high level of parallelism is obtained at every iteration.
(iii) *Cheap exchange.* Global communications of agents with a central node are cheap (scalars). More costly communications (vectors) remain local between neighbouring agents. In general, the amount of communication should be kept as low as possible. It is already clear that globalisation strategies based on line-search do not fit with the distributed framework (Fei, Y. and Guodong, R. and Wang, B. and Wang, W., 2014), as these entail evaluating a 'central' merit function multiple times per iteration, thus significantly increasing communications.
(iv) *Inexactness.* Convergence is 'robust' to inexact solutions of the subproblems, since it may be necessary to truncate the number of sub-iterations due to communication costs.
(v) *Fast convergence.* The sequence of iterates converges at a fast (at least linear) local rate. Slow convergence generally results in a prohibitively high number of communications.

As we are interested in applications such as DNMPC, which require solving distributed parametric NLPs with a low latency (Hours, J.-H. and Jones, C.N., 2016), a desirable feature of our algorithm should also be

(vi) *Warm-start and activity detection.* The algorithm detects the optimal active-set quickly and enables warm-starting.

Whereas a fair number of well-established algorithms exist for solving distributed convex NLPs (Bertsekas, D.P. and Tsitsiklis, J.N., 1997), there is, as yet, no consensus around a set of practical methods applicable to distributed nonconvex programs. Some work (Zavala, V.M. and Laird, C.D. and Biegler, L.T., 2008) exists on the parallelisation of linear algebra operations involved in solving nonconvex NLPs with IPOPT (Wächter, A. and Biegler, L.T., 2006), but the approach is limited to very specific problem structures and the globalisation phase of IPOPT (filter line-search) is not suitable for fully distributed implementations (requirements (iii), (iv) and (vi) are not met). Among existing strategies capable of addressing a broader class of distributed nonconvex programs, one can make a clear distinction between Sequential Convex Programming (SCP) approaches and augmented Lagrangian techniques.

An SCP method consists in iteratively solving distributed convex NLPs, which are local approximations of the original nonconvex NLP. To date, some of the most efficient algorithms for solving distributed convex NLPs combine dual decomposition with smoothing techniques (Necoara, I. and

Savorgnan, C. and Tran Dinh, Q. and Suykens, J. and Diehl, M., 2009; Tran-Dinh, Q. and Savorgnan, C. and Diehl, M., 2013). On the contrary, an augmented Lagrangian method aims at decomposing a nonconvex auxiliary problem inside an augmented Lagrangian loop (Cohen, G., 1980; Hamdi, A. and Mishra, S.K., 2011; Hours, J.-H. and Jones, C.N., 2014). While convergence guarantees can be derived in both frameworks, computational drawbacks also exist on both sides. For instance, it is not clear how to preserve the convergence properties of SCP schemes when every subproblem is solved to a low level of accuracy. Hence, (iv) is not satisfied immediately. Nevertheless, for some recent work in this direction, one may refer to (Tran Dinh, Q. and Necoara, I. and Diehl, M., 2013). The convergence rate of the algorithm analysed in (Tran Dinh, Q. and Necoara, I. and Diehl, M., 2013) is at best linear, thus not fulfilling (v). On the contrary, the inexactness issue can be easily handled inside an augmented Lagrangian algorithm, as global and fast local convergence is guaranteed even though the subproblems are not solved to a high level of accuracy (Fernández, D. and Solodov, M.V., 2012; Conn, A. and Gould, N.I.M. and Toint, P.L., 1991). However, in practice, poor initial estimates of the dual variables can drive the iterative process to infeasible points. Moreover, it is still not clear how the primal nonconvex subproblems should be decomposed and solved efficiently in a distributed context. The quadratic penalty term of an augmented Lagrangian does not allow for the same level of parallelism as a (convex) dual decomposition. Thus, requirement (ii) is not completely satisfied. To address this issue, we have recently proposed applying Proximal Alternating Linearised Minimisations (PALM) (Bolte, J. and Sabach, S. and Teboulle, M., 2014) to solve the auxiliary augmented Lagrangian subproblems (Hours, J.-H. and Jones, C.N., 2014,6). The resulting algorithm inherits the slow convergence properties of proximal gradient methods and does not readily allow one to apply a preconditioner. In this paper, a novel mechanism for handling the augmented Lagrangian subproblems in a more efficient manner is proposed and analysed. The central idea is to use alternating gradient projections to compute a Cauchy point in a trust region Newton method (Conn, A.R. and Gould, N.I.M. and Toint, P.L., 2000).

When looking at practical trust region methods for solving bound-constrained problems (Zavala, V.M. and Anitescu, M., 2014), one may notice that the safeguarded Conjugate Gradient (sCG) algorithm is well-suited to distributed implementations, as the main computational tasks are structured matrix-vector and vector-vector multiplications, which do not require the assembly of a matrix on a central node. Moreover, the global communications involved in an sCG algorithm are cheap. Thus, sCG satisfies (i), (ii) and (iii). The implementation of CG on distributed architectures has been extensively explored (Verschoor, M. and Jalba, A.C., 2012; D'Azevedo, E. and Eijkhout, V. and Romine, C., 1993; Fei, Y. and Guodong, R. and Wang, B. and Wang, W., 2014). Furthermore, a trust region update requires only one centralized objective evaluation per iteration. From a computational perspective, it is thus comparable to a dual update, which requires evaluating the constraints functional and is ubiquitous in distributed optimisation algorithms. However, computing the Cauchy point in a trust region loop is generally done by means of a projected line-search (Zavala, V.M. and Anitescu, M., 2014) or sequential search (Conn, A.R. and Gould, N.I.M and Toint, P.L., 1988). Whereas it is broadly admitted that the Cauchy point computation is cheap, this operation requires a significant amount of global communications in distributed memory parallel environments, and is thus hardly amenable to such applications (Fei, Y. and Guodong, R. and Wang, B. and Wang, W., 2014). This hampers the implementability of trust region methods with good convergence guarantees on distributed computing platforms, whereas many parts of the algorithm are attractive for such implementations. The aim of this paper is to bridge the gap by proposing a novel way of computing the Cauchy point that is more tailored to the distributed framework. Coordinate gradient descent methods such as PALM, are known to be parallelisable for some partial separability structures (Bertsekas, D.P. and Tsitsiklis, J.N., 1997). Moreover, in practice, the number of backtracking iterations necessary to select a block step-size, can be easily bounded, making the approach suitable for 'Same Instruction Multiple Data' architectures. Therefore, we propose using one sweep of block-coordinate gradient descent to compute a Cauchy point. As shown in paragraph 4.2 of Section 4, such a strategy turns out to be efficient at identifying the optimal active-set. It can then be accelerated by means of an inexact Newton method. As our algorithm differs from the usual trust region Newton method, we provide a detailed convergence analysis in Section 4. Finally, one should mention a recent paper (Xue, D. and Sun, W. and Qi, L., 2014), in which a trust region method is combined with alternating minimisations, namely the Alternating Directions Method of Multipliers (ADMM) (Bertsekas, D.P. and Tsitsiklis, J.N., 1997), but in a very different way from the strategy described next. The contributions of the paper are the following:

- We propose a novel way of computing a Cauchy point in a trust region framework, which is suitable for distributed implementations.
- We adapt the standard trust region algorithm to the proposed Cauchy point computation. Global convergence along with an almost Q-superlinear local rate is proven under standard assumptions.
- The proposed trust region algorithm, entitled TRAP (Trust Region with Alternating Projections), is used as a primal solver in an augmented Lagrangian dual loop, resulting in an algorithm that meets requirements (i)-(vi), and is applied to solve OPF programs in a distributed fashion.

In Section 2, some basic notion in variational analysis is recalled. In Section 3, our TRAP algorithm is presented. Its convergence properties are analysed in Section 4. Global convergence to a critical point is guaranteed, as well as almost Q-superlinear local convergence. Finally, the proposed algorithm is tested on OPF problems over transmission and distribution networks in Section 5.

2 Background

Given a closed convex set Ω, the projection operator onto Ω is denoted by P_{Ω} and the indicator function of Ω is defined by

$$\iota_{\Omega}\left(x\right)=\begin{cases}0\ , & \text{if } x\in\Omega\ ,\\ +\infty\ , & \text{if } x\notin\Omega\ .\end{cases}$$

We define the normal cone to Ω at $x\in\Omega$ as

$$\mathcal{N}_{\Omega}\left(x\right):=\left\{v\in\mathbb{R}^{d}\ :\ \forall y\in\Omega,\langle v,y-x\rangle\leq 0\right\}\ .$$

The tangent cone to Ω at x is defined as the closure of feasible directions at x (Rockafellar, R.T. and Wets, R.J.-B., 2009). Both $\mathcal{N}_{\Omega}\left(x\right)$ and $\mathcal{T}_{\Omega}\left(x\right)$ are closed convex cones. As Ω is convex, for all $x\in\Omega$, $\mathcal{N}_{\Omega}\left(x\right)$ and $\mathcal{T}_{\Omega}\left(x\right)$ are polar to each other (Rockafellar, R.T. and Wets, R.J.-B., 2009).

Theorem 21 (Moreau's decomposition (Moreau, 1962))**.** *Let $\mathcal{K}$ be a closed convex cone in $\mathbb{R}^d$ and $\mathcal{K}^{\circ}$ its polar cone. For all $x,y,z\in\mathbb{R}^d$, the following two statements are equivalent:*

1. $z=x+y$ with $x\in\mathcal{K}$, $y\in\mathcal{K}^{\circ}$ and $\langle x,y\rangle=0$.
2. $x=P_{\mathcal{K}}\left(z\right)$ and $y=P_{\mathcal{K}^{\circ}}\left(z\right)$.

The box-shaped set $\left\{x\in\mathbb{R}^{d}\ :\ \forall i\in\left\{1,\ldots,d\right\},l_i\leq x_i\leq u_i\right\}$ is denoted by $\mathbb{B}\left(l,u\right)$. For $x\in\mathbb{R}^d$ and $r>0$, the open ball of radius r centered around x is denoted by $\mathcal{B}\left(x,r\right)$. Given $x\in\Omega$, the set of active constraints at x is denoted by $\mathcal{A}_{\Omega}\left(x\right)$. Given a set $S\subseteq\mathbb{R}^d$, its relative interior is defined as the interior of S within its affine hull, and is denoted by $\mathrm{ri}\left(S\right)$.

A critical point x^* of the function $f+\iota_{\Omega}$ with f differentiable, is said to be non-degenerate if

$$-\nabla f\left(x^*\right)\in\mathrm{ri}\left(\mathcal{N}_{\Omega}\left(x^*\right)\right)\ .$$

Given a differentiable function f of several variables $x_1, \dots, x_N$, its gradient with respect to variable x_i is denoted by $\nabla_i f$. Given a matrix $M \in \mathbb{R}^{m \times n}$, its (i, j) element is denoted by $M_{i,j}$.

A sequence $\left\{x^l\right\}$ converges to x^* at a Q-linear rate $\varrho \in]0,1[$ if, for l large enough,

$$\frac{\left\|x^{l+1} - x^*\right\|_2}{\left\|x^l - x^*\right\|_2} \leq \varrho \ .$$

The convergence rate is said to be Q-superlinear if the above ratio tends to zero as l goes to infinity.

3 A Trust Region Algorithm with Distributed Activity Detection

3.1 Algorithm Formulation

The problem we consider is that of minimising a partially separable objective function subject to separable convex constraints.

$$\begin{aligned} &\underset{w}{\text{minimise}} \ L\left(w_1, \dots, w_N\right) \\ &\text{s.t. } w_i \in \mathcal{W}_i, \ \forall i \in \{1, \dots, N\} \ \ , \end{aligned} \tag{1}$$

where $w := \left(w_1^\top, \dots, w_N^\top\right)^\top \in \mathbb{R}^n$, with $n = \sum_{i=1}^N n_i$, and $\mathcal{W} := \mathcal{W}_1 \times \ldots \times \mathcal{W}_N$, where the sets $\mathcal{W}_i \subset \mathbb{R}^{n_i}$ are closed and convex. The following Assumption is standard in distributed computations (Bertsekas, D.P. and Tsitsiklis, J.N., 1997).

Assumption 31 (Colouring scheme)**.** *The sub-variables $w_1, \dots, w_N$ can be re-ordered and grouped together in such a way that a Gauss-Seidel minimisation sweep on the function L can be performed in parallel within $K \ll N$ groups, which are updated sequentially. In the sequel, the re-ordered variable is denoted by $x = \left(x_1^\top, \dots, x_K^\top\right)^\top$. The set $\mathcal{W}$ is transformed accordingly into $\Omega = \Omega_1 \times \ldots \times \Omega_K$. It is worth noting that each set Ω_k with $k \in \{1, \dots, K\}$ can then be decomposed further into sets $\mathcal{W}_i$ with $i \in \{1, \dots, N\}$.*

As a consequence of Assumption 31, NLP (1) is equivalent to

$$\begin{aligned} &\underset{x}{\text{minimise}} \ L\left(x_1, \dots, x_K\right) \\ &\text{s.t. } x_k \in \Omega_k, \ \forall k \in \{1, \dots, K\} \ \ . \end{aligned}$$

Remark 31. *Such a partially separable structure in the objective (Assumption 31) is encountered very often in practice, for instance when relaxing network coupling constraints via an augmented Lagrangian penalty. Thus, by relaxing the nonlinear coupling constraint $C\left(w_1, \dots, w_N\right) = 0$ and the local equality constraints $g_i\left(w_i\right) = 0$ of*

$$\begin{aligned} \underset{w_1, \dots, w_N}{\text{minimise}} &\sum_{i=1}^N f_i\left(w_i\right) \\ s.t. \ C\left(w_1, \dots, w_N\right) &= 0 \\ g_i\left(w_i\right) &= 0 \\ w_i &\in \mathcal{W}_i \\ i &\in \{1, \dots, N\} \ \ , \end{aligned}$$

in a differentiable penalty function, one obtains an NLP of the form (1)*. In NLPs resulting from the direct transcription of optimal control problems, the objective is generally separable and the constraints are stage-wise with a coupling between the variables at a given time instant with the variables of the next time instant. In this particular case, the number of groups is* $K=2$*. In Section 5, we illustrate this property by means of examples arising from various formulations of the Optimal Power Flow (OPF) problem. The number of colours* K *represents the level of parallelism that can be achieved in a Gauss-Seidel method for solving* (1)*. Thus, in the case of a discretised OCP, an alternating projected gradient sweep can be applied in two steps during which all updates are parallel.*

For the sake of exposition, in order to make the distributed nature of our algorithm apparent, we assume that every sub-variable w_i, with $i \in \{1,\ldots,N\}$, is associated with a computing node. Two nodes are called *neighbours* if they are coupled in the objective L. Our goal is to find a first-order critical point of NLP (1) via an iterative procedure for which we are given an initial feasible point $x^0 \in \Omega$. The iterative method described next aims at computing every iterate in a distributed fashion, which requires communications between neighbouring nodes and leads to a significant level of concurrency.

Assumption 32. *The objective function* L *is bounded below on* $\left\{x \in \Omega \;:\; L(x) \leq L(x^0)\right\}$*.*

The algorithm formulation can be done for any convex set Ω, but some features are more suitable for linear inequality constraints.

Assumption 33 (Polyhedral constraints)**.** *For all* $k \in \{1,\ldots,K\}$*, the set* Ω_k *is a non-empty polyhedron, such that*

$$\Omega_k := \{x \in \mathbb{R}^{n_k} \;:\; \langle \omega_{k,i}, x\rangle \leq h_{k,i},\; i \in \{1,\ldots,m_k\}\} \enspace ,$$

with $\omega_{k,i} \in \mathbb{R}^{n_k}$*,* $h_{k,i} \in \mathbb{R}$ *for all* $i \in \{1,\ldots,m_k\}$ *and* $n_k, m_k \geq 1$*.*

Assumption 34. *The objective function* L *is continuously differentiable in an open set containing* Ω*. Its gradient* ∇L *is uniformly continuous.*

It is well-known (Conn, A.R. and Gould, N.I.M. and Toint, P.L., 2000) that for problem (1), x^* being a critical point is equivalent to

$$P_\Omega\left(x^* - \nabla L\left(x^*\right)\right) = x^* \enspace . \tag{2}$$

Algorithm 1 below is designed to compute a critical point x^* of the function $L + \iota_\Omega$. It is essentially a two-phase approach, in which an active-set is first computed and then, a quadratic model is minimised approximately on the current active face. Standard two-phase methods compute the active-set by means of a centralised projected search, updating all variables centrally. More precisely, a model of the objective is minimised along the projected objective gradient, which yields the Cauchy point. The model decrease provided by the Cauchy point is then enhanced in a refinement stage. Similarly to a two-phase method, in order to globalise convergence, Algorithm 1 uses the standard trust region mechanism. At every iteration, a model m of the objective function L is constructed around the current iterate x as follows

$$m\left(x'\right) := L\left(x\right) + \left\langle \nabla L\left(x\right), x' - x\right\rangle + \frac{1}{2}\left\langle x' - x, B\left(x\right)\left(x' - x\right)\right\rangle \enspace , \tag{3}$$

where $x' \in \mathbb{R}^n$ and $B\left(x\right)$ is a symmetric matrix.

Assumption 35 (Uniform bound on model hessian)**.** *There exists* $\hat{B} > 0$ *such that*

$$\left\|B\left(x\right)\right\|_2 \leq \hat{B} \enspace ,$$

for all $x \in \Omega$*.*

Algorithm 1 Trust Region Algorithm with Alternating Projections (trap)

```
1: Parameters: Initial trust region radius $\Delta$, update parameters $\sigma_1$, $\sigma_2$ and $\sigma_3$ such that $0 < \sigma_1 < \sigma_2 < 1 < \sigma_3$,
   test ratios $\eta_1$ and $\eta_2$ such that $0 < \eta_1 < \eta_2 < 1$, coefficients $\gamma_1 \in ]0,1[$ and $\gamma_2 > 0$, termination tolerance $\epsilon$.
2: Input: Initial guess $x$, projection operators $\left\{P_{\Omega_k}\right\}_{k=1}^{K}$, objective function $L$, objective gradient $\nabla L$.
3: while $\left\|P_\Omega\left(x - \nabla L\left(x\right)\right) - x\right\|_2 > \epsilon$ do
4:     Distributed activity detection (alternating gradient projections):
5:     for $k = 1 \ldots, K$ do
6:         $z_k \leftarrow P_{\Omega_k}\left(x_k - \alpha_k \nabla_k m\left(z_{[\![1,k-1]\!]}, x_k, x_{[\![k+1,K]\!]}\right)\right)$,        ▷ In parallel in group $k$
7:         where $\alpha_k$ is computed according to requirements (5), (6) and (7).
8:     end for
9:     Distributed refinement (Algorithm 2):
10:    Find $y \in \Omega$ such that
11:        $m\left(x\right) - m\left(y\right) \geq \gamma_1\left(m\left(x\right) - m\left(z\right)\right)$
12:        $\left\|y - x\right\|_\infty \leq \gamma_2 \Delta$
13:        $\mathcal{A}_{\Omega_k}\left(z_k\right) \subset \mathcal{A}_{\Omega_k}\left(y_k\right)$ for all $k \in \{1, \ldots, K\}$.
14:    Trust-region update:
15:    $\rho \leftarrow {}^{L(x)-L(y)}/_{m(x)-m(y)}$
16:    if $\rho < \eta_1$ then                                        ▷ Not successful
17:        (Do not update x)
18:        Take $\Delta$ within $[\sigma_1 \Delta, \sigma_2 \Delta]$
19:    else if $\rho \in [\eta_1, \eta_2]$ then                          ▷ Successful
20:        $x \leftarrow y$
21:        Take $\Delta$ within $[\sigma_1 \Delta, \sigma_3 \Delta]$
22:        Update objective gradient $\nabla L\left(x\right)$ and model hessian $B\left(x\right)$.
23:    else                                                      ▷ Very successful
24:        $x \leftarrow y$
25:        Take $\Delta$ within $[\Delta, \sigma_3 \Delta]$
26:        Update objective gradient $\nabla L\left(x\right)$ and model hessian $B\left(x\right)$
27:    end if
28: end while
```

The following Assumption is necessary to ensure distributed computations in Algorithm 1. It is specific to Algorithm 1 and does not appear in the standard trust region methods (Burke et al, 1990).

Assumption 36 (Structured model hessian)**.** *For all $x \in \Omega$, for all $i, j \in \{1, \ldots, n\}$,*

$$\nabla^2 L_{i,j}\left(x\right) = 0 \Leftrightarrow B_{i,j}\left(x\right) = 0 \ .$$

Remark 32. *It is worth noting that the partial separability structure of the objective function L is transferred to the sparsity pattern of its hessian $\nabla^2 L$, hence, by Assumption 36, to the sparsity pattern of the model hessian B. Hence, a Gauss-Seidel sweep on the model function m can also be carried out in K parallel steps.*

The main characteristic of trap is the activity detection phase, which differs from the projected search in standard trust region methods (Burke et al, 1990). At every iteration, trap updates the current active-set by computing iterates $z_1, \ldots, z_K$ (Lines 4 to 8). This is the main novelty of trap, compared to existing two-phase techniques, and allows for different step-sizes $\alpha_1, \ldots, \alpha_K$ per block of variables, which is relevant in a distributed framework, as the current active-set can be split among nodes and does not need to be computed centrally. In the trust region literature, the point

$$z := \left(z_1^\top, \ldots, z_K^\top\right)^\top \ , \tag{4}$$

is often referred to as the *Cauchy point*. We keep this terminology in the remainder of the paper. It is clear from its formulation that trap allows one to compute Cauchy points via independent projected searches on every node. Once the Cauchy points $z_1, \ldots, z_K$ have been computed, they are used in the refinement step to compute a new iterate y that satisfies the requirements shown from Lines 9 to 13. The last step consists in checking if the model decrease $m(y) - m(x)$ is close enough to the variation in the objective L (Lines 16 to 27). In this case, the iterate is updated and the trust region radius $\mathit{\Delta}$ increased, otherwise the radius is shrunk and the iterate frozen. This operation requires a global exchange of information between nodes.

In the remainder, the objective gradient $\nabla L\left(x\right)$ is denoted by $g\left(x\right)$ and the objective hessian $\nabla^2 L\left(x\right)$ by $H\left(x\right)$. The model function m is an approximation of the objective function L around the current iterate x. The quality of the approximation is controlled by the trust region, defined as the box

$$\mathbb{B}\left(x - \mathit{\Delta}, x + \mathit{\Delta}\right) \ \ ,$$

where $\mathit{\Delta}$ is the trust region radius.

In the rest of the paper, we denote the Cauchy points by z_k or $z_k\left(\alpha_k\right)$ without distinction, where α_k are appropriately chosen step-sizes. More precisely, following Section 3 in (Burke et al, 1990), in TRAP, the block-coordinate step-sizes α_k are chosen so that for all $k \in \{1, \ldots, K\}$, the Cauchy points z_k satisfy

$$\begin{cases} m\left(z_{[\![1,k-1]\!]}, z_k, x_{[\![k+1,K]\!]}\right) \leq m\left(z_{[\![1,k-1]\!]}, x_k, x_{[\![k+1,K]\!]}\right) \\ \qquad\qquad\qquad\qquad + \nu_0 \left\langle \nabla_k m\left(z_{[\![1,k-1]\!]}, x_k, x_{[\![k+1,K]\!]}\right), z_k - x_k \right\rangle \ \ , \\ \left\|z_k - x_k\right\|_\infty \leq \nu_2 \mathit{\Delta} \end{cases} \tag{5}$$

with $\nu_0 \in \,]0,1[$ and $\nu_2 > 0$, where $z_{[\![1,k-1]\!]}$ stands for $\left(z_1^\top, \ldots, z_{k-1}^\top\right)^\top$, along with the condition that there exists positive scalars $\nu_1 < \nu_2$, ν_3, ν_4 and ν_5 for all $k \in \{1, \ldots, K\}$,

$$\alpha_k \in [\nu_4, \nu_5] \qquad \text{or} \qquad \alpha_k \in [\nu_3 \bar{\alpha}_k, \nu_5] \tag{6}$$

where the step-sizes $\bar{\alpha}_k$ are such that one of the following conditions hold for every $k \in \{1, \ldots, K\}$,

$$\begin{aligned} m\left(z_{[\![1,k-1]\!]}, z_k\left(\bar{\alpha}_k\right), x_{[\![k+1,K]\!]}\right) > {} & m\left(z_{[\![1,k-1]\!]}, x_k, x_{[\![k+1,K]\!]}\right) \\ & + \nu_0 \left\langle \nabla_k m\left(z_{[\![1,k-1]\!]}, x_k, x_{[\![k+1,K]\!]}\right), z_k\left(\bar{\alpha}_k\right) - x_k \right\rangle \ \ , \end{aligned} \tag{7}$$

or

$$\left\|z_k\left(\bar{\alpha}_k\right) - x_k\right\|_\infty \geq \nu_1 \mathit{\Delta} \ \ , \tag{8}$$

Conditions (5) ensure that the step-sizes α_k are small enough to enforce a sufficient decrease coordinate-wise, as well as containment within a scaled trust region. Conditions (6), (7) and (8) guarantee that the step-sizes α_k do not become arbitrarily small. All conditions (5), (6) and (7) can be tested in parallel in each of the K groups of variables. In the next two paragraphs 3.2 and 3.3 of this section, the choice of step-sizes α_k ensuring the sufficient decrease is clarified, as well as the distributed refinement step. In the next Section 4, the convergence properties of TRAP are analysed. Numerical examples are presented in Section 5.

3.2 Step-sizes Computation in the Activity Detection Phase

At a given iteration of TRAP, the step-sizes α_k are computed by backtracking to ensure a sufficient decrease at every block of variables and coordinate-wise containment in a scaled trust region as formalised by (5). It is worth noting that the coordinate-wise backtracking search can be run in parallel among the variables of group k, as they are decoupled from each other. As a result, there is one step-size per subvariable w_i in group k. Yet, for simplicity, we write it as a single step-size α_k. The reasoning of Section 4 can be adapted accordingly. The following Lemma shows that a coordinate-wise step-size α_k can be computed that ensures conditions (5), (6), (7) and (8) on every block of coordinates $k \in \{1, \ldots, K\}$.

Lemma 31. *Assume that Assumption 35 holds. For all $k \in \{1, \ldots, K\}$, an iterate z_k satisfying conditions* (5)*,* (6)*,* (7) *and* (8) *can be found after a finite number of backtracking iterations.*

Proof. Let $k \in \{1, \ldots, K\}$. We first show that for a sufficiently small α_k, conditions (5) are satisfied. By definition of the Cauchy point z_k,

$$z_k = \underset{z \in \Omega_k}{\operatorname{argmin}} \ \left\langle \nabla_k m \left(z_{[\![1,k-1]\!]}, x_k, x_{[\![k+1,K]\!]} \right), z - x_k \right\rangle + \frac{1}{2\alpha_k} \left\| z - x_k \right\|_2^2 \ ,$$

which implies that

$$\left\langle \nabla_k m \left(z_{[\![1,k-1]\!]}, x_k, x_{[\![k+1,K]\!]} \right), z_k - x_k \right\rangle + \frac{1}{2\alpha_k} \left\| z_k - x_k \right\|_2^2 \leq 0 \ ,$$

Hence, as $\nu_0 \in]0, 1[$, it follows that

$$\begin{aligned} \left\langle \nabla_k m \left(z_{[\![1,k-1]\!]}, x_k, x_{[\![k+1,K]\!]} \right), z_k - x_k \right\rangle + \frac{1-\nu_0}{2\alpha_k} \left\| z_k - x_k \right\|_2^2 \leq \\ \nu_0 \left\langle \nabla_k m \left(z_{[\![1,k-1]\!]}, x_k, x_{[\![k+1,K]\!]} \right), z_k - x_k \right\rangle \ . \end{aligned}$$

However, from the descent Lemma, which can be applied since the model gradient is Lipschitz continuous by Assumption 35,

$$\begin{aligned} m \left(z_{[\![1,k-1]\!]}, z_k, x_{[\![k+1,K]\!]} \right) \leq m \left(z_{[\![1,k-1]\!]}, x_k, x_{[\![k+1,K]\!]} \right) + \left\langle \nabla_k m \left(z_{[\![1,k-1]\!]}, x_k, x_{[\![k+1,K]\!]} \right), z_k - x_k \right\rangle \\ + \frac{\hat{B}}{2} \left\| z_k - x_k \right\|_2^2 \ . \end{aligned}$$

By choosing

$$\alpha_k \leq \frac{1-\nu_0}{\hat{B}} \ ,$$

condition (5) is satisfied after a finite number of backtracking iterations. Denoting by q_k the smallest integer such that requirement (5) is met, α_k can be written

$$\alpha_k = c^{q_k} \cdot \alpha^{(0)} \ ,$$

where $c \in]0, 1[$ and $\alpha^{(0)} > 0$. Then, condition (6) is satisfied with $\nu_4 = \alpha^{(0)}$ and $\nu_3 = c$. □

Lemma 31 is very close to Theorem 4.2 in (Moré, J.J., 1988), but the argument regarding the existence of the step-sizes α_k is different.

3.3 Distributed Computations in the Refinement Step

In Algorithm 1, the objective gradient $g(x)$ and model hessian $B(x)$ are updated after every successful iteration. This task requires exchanges of variables between neighbouring nodes, as the objective is partially separable (Ass. 31). Node i only needs to store the sub-part of the objective function L that combines its variable w_i and the variables associated to its neighbours. However, the refinement step (line 9 to 13 in Algorithm 1), in which one obtains a fraction of the model decrease yielded by the Cauchy points $z_1, \ldots, z_K$, should also be computed in a distributed manner. As detailed next, this phase consists in solving the Newton problem on the subspace of free variables at the current iteration, which is defined as the set of free variables at the Cauchy points $z_1, \ldots, z_K$. In order to achieve a reasonable level of efficiency in the trust region procedure, this step is generally performed via the Steihaug-Toint CG, or sCG (Steihaug, T., 1983). The sCG algorithm is a CG procedure that is cut if a negative curvature direction is encountered or a problem bound is hit in the process. Another way of improving on the Cauchy point to obtain fast local convergence is the Dogleg strategy (Nocedal, J. and Wright, S., 2006). However, this technique requires the model hessian B to be positive definite (Nocedal, J. and Wright, S., 2006). This condition does not fit well with distributed computations, as positive definiteness is typically enforced by means of BFGS updates, which are know for not preserving the sparsity structure of the objective without non-trivial modifications and assumptions (Yamashita, N., 2008). Compared to direct methods, iterative methods such as the sCG procedure have clear advantages in a distributed framework, for they do not require assembling the hessian matrix on a central node. Furthermore, their convergence speed can be enhanced via block-diagonal preconditioning, which is suitable for distributed computations. In the sequel, we briefly show how a significant level of distributed operations can be obtained in the sCG procedure, mainly due to the sparsity structure of the model hessian that matches the partial separability structure of the objective function. More details on distributed implementations of the CG algorithm can be found in numerous research papers (Verschoor, M. and Jalba, A.C., 2012; D'Azevedo, E. and Eijkhout, V. and Romine, C., 1993; Fei, Y. and Guodong, R. and Wang, B. and Wang, W., 2014). The sCG algorithm that is described next is a rearrangement of the standard sCG procedure following the idea of (D'Azevedo, E. and Eijkhout, V. and Romine, C., 1993). The two separate inner products that usually appear in the CG are grouped together at the same stage of the Algorithm.

An important feature of the refinement step is the increase of the active set at every iteration. More precisely, in order to ensure finite detection of activity, the set of active constraints at the points $y_1, \ldots, y_K$, obtained in the refinement phase, needs to contain the set of active constraints at the Cauchy points $z_1, \ldots, z_K$, as formalised at line 13 of Algorithm 1. This requirement is very easy to fulfil when Ω is a bound constraint set, as it just requires enforcing the constraint

$$y_{k,i} = z_{k,i}, \ i \in \left\{ j \in \{1, \ldots, n_k\} \ : \ z_{k,j} = \underline{x}_{k,j} \text{ or } \bar{x}_{k,j} \right\}$$

for all groups $k \in \{1, \ldots, K\}$ in the trust region problem at the refinement step.

For the convergence analysis that follows in Section 4, the refinement step needs to be modified compared to existing trust region techniques. Instead of solving the standard refinement problem

$$\begin{aligned} \underset{p}{\text{minimise}} \ & \langle g(x), p \rangle + \frac{1}{2} \langle p, B(x) p \rangle \\ \text{s.t.} \ & \|p\|_{\infty} \leq \gamma_2 \Delta \\ & x + p \in \Omega \\ & \mathcal{A}_{\Omega}(z) \subseteq \mathcal{A}_{\Omega}(x+p) \quad , \end{aligned}$$

in which the variables corresponding to indices of active constraints at the Cauchy point z are fixed to zero, we solve a regularised version

$$\begin{aligned} \underset{y\in\Omega}{\text{minimise}} \;\; & \langle g\left(x\right), y-x\rangle + \frac{1}{2}\langle y-x, B\left(x\right)\left(y-x\right)\rangle + \frac{\sigma}{2}\left\|y-z\right\|_2^2 \\ \text{s.t.} \;\; & \left\|y-x\right\|_\infty \leq \gamma_2\Delta \\ & \mathcal{A}_\Omega\left(z\right) \subseteq \mathcal{A}_\Omega\left(y\right) \quad , \end{aligned} \tag{9}$$

where $\sigma \in \left]\underline{\sigma}, \bar{\sigma}\right[$ with $\underline{\sigma} > 0$, and z is the Cauchy point yielded by the procedure described in the previous paragraph 3.2. The regularisation coefficient σ should not be chosen arbitrarily, as it may inhibit the fast local convergence properties of the Newton method. This point is made explicit in paragraph 4.3 of Section 4. The regularised trust region subproblem (9) can be equivalently written

$$\begin{aligned} \underset{p}{\text{minimise}} \;\; & \langle g_\sigma\left(x\right), p\rangle + \frac{1}{2}\langle p, B_\sigma\left(x\right)p\rangle \\ \text{s.t.} \;\; & x+p \in \Omega \\ & \left\|p\right\|_\infty \leq \gamma_2\Delta \\ & \mathcal{A}_\Omega\left(z\right) \subseteq \mathcal{A}_\Omega\left(x+p\right) \quad , \end{aligned} \tag{10}$$

with

$$g_\sigma\left(x\right) := g\left(x\right) - \sigma(z-x), \quad B_\sigma\left(x\right) := B\left(x\right) + \frac{\sigma}{2}I \quad . \tag{11}$$

As in standard trust region methods, we solve the refinement subproblem (10) by means of CG iterations, which can be distributed as a result of Assumption 36. In order to describe this stage in Algorithm 2, one needs to assume that Ω is a box constraint set. In the remainder, we denote by Z the matrix whose columns are an orthonormal basis of the subspace

$$V\left(z\right) := \left\{x \in \mathbb{R}^n \;:\; \langle\omega_{k,i}, x_k\rangle = 0,\; i \in \mathcal{A}_{\Omega_k}\left(z_k\right),\; k \in \{1,\ldots,K\}\right\} \quad .$$

Remark 33. *It is worth noting that the requirement $m(x) - m(y) \geq \gamma_1\left(m(x) - m(z)\right)$, with $\gamma_1 < 1$, is satisfied after any iteration of Algorithm 2, as the initial guess is the Cauchy point z and the sCG iterations ensure monotonic decrease of the regularised model (Theorem* 2.1 *in (Steihaug, T.,* 1983*)).*

Remark 34. *It is worth noting that the sparsity pattern of the reduced model hessian $\hat{B}_\sigma$ has the same structure as the sparsity pattern of the model hessian B, as the selection matrix Z has a block-diagonal structure. Moreover, the partial separability structure of the objective matches the sparsity patterns of both the hessian and the reduced hessian. For notational convenience, Algorithm 2 is written in terms of variables $x_1, \ldots, x_K$, but it is effectively implementable in terms of variables $w_1, \ldots, w_N$. The inner products (Lines 6 to 8) and updates (Lines 11 to 14, lines 20 to 22) can be computed in parallel at every node, as well as the structured matrix-vector product (Line 5).*

In Algorithm 2, the reduced model hessian $\hat{B}$ can be evaluated when computing the product at line 5, which requires local exchanges of vectors between neighbouring nodes, since the sparsity pattern of $\hat{B}$ represents the coupling structure in the objective L. From a distributed implementation perspective, the more costly parts of the refinement procedure 2 are at line 9 and line 16. These operations consist in summing up the inner products from all nodes and a minimum search over the step-sizes that ensure constraint satisfaction and containment in the trust region. They need to be performed on a central node that has access to all data from other nodes, or via a consensus algorithm. Therefore,

Algorithm 2 Distributed Safeguarded Conjugate Gradient (sCG)

```
1: **Input:** reduced model hessian $\hat{B}_\sigma := Z^\top B_\sigma Z$, reduced gradient $\hat{g} := Z^\top g$, initial guess $\hat{z} := Z^\top z$
2: **Parameters:** stopping tolerance $\hat{\epsilon} := \xi \left\| \hat{g} \right\|_2$ with $\xi \in ]0,1[$
3: Initialise $\hat{x}$, $\hat{p}$, $\hat{v}$, $\hat{r}$, $\hat{t}$ and $\hat{u}_{\text{prev}}$ via a standard sCG iteration using $z$, $x$, $Z$, $B_\sigma$, $\hat{B}_\sigma$ and $\hat{g}$
4: **while** $\hat{u} > \hat{\epsilon}^2$ and $\hat{t} > 0$ **do**
5:     Compute structured matrix-vector product $\hat{s} \leftarrow \hat{B}_\sigma \hat{r}$        ▷ Local communications
6:     **for** $k = 1 \ldots K$ **do**        ▷ In parallel among $K$ groups
7:         Compute $\langle \hat{r}_k, \hat{r}_k \rangle$ and $\langle \hat{r}_k, \hat{s}_k \rangle$
8:     **end for**
9:     $\hat{u} \leftarrow \sum_{i=k}^{K} \langle \hat{r}_k, \hat{r}_k \rangle$, $\hat{\delta} \leftarrow \sum_{k=1}^{K} \langle \hat{r}_k, \hat{s}_k \rangle$        ▷ Global summations
10:    Compute step-sizes $\hat{\beta} \leftarrow \hat{u}/\hat{u}_{\text{prev}}$ and $\hat{t} \leftarrow \hat{\delta} - \hat{\beta}^2 \hat{t}$
11:    **for** $k = 1 \ldots K$ **do**        ▷ In parallel among $K$ groups
12:        Update conjugate direction $\hat{p}_k \leftarrow \hat{r}_k + \hat{\beta}\hat{p}_k$ and $\hat{v}_k \leftarrow \hat{s}_k + \hat{\beta}\hat{v}_k$
13:        Compute smallest step-size $a_k$ such that $\hat{x}_k + a_k \hat{p}_k$ hits a bound $\underline{x}_k$, $\bar{x}_k$ or the trust region boundary
14:    **end for**
15:    **if** $\hat{t} \leq 0$ **then**        ▷ Negative curvature check
16:        Compute step-size $\hat{a} \leftarrow \min\{a_1, \ldots, a_K\}$ to hit boundary of $\mathbb{B}(x - \Delta, x + \Delta) \cap \Omega$
17:    **else**
18:        Compute standard CG step-size $\hat{a} \leftarrow \hat{u}/\hat{t}$
19:    **end if**
20:    **for** $k = 1 \ldots K$ **do**        ▷ In parallel among $K$ groups
21:        Update iterate $\hat{x}_k \leftarrow \hat{x}_k + \hat{a}\hat{p}_k$ and residual $\hat{r}_k \leftarrow \hat{r}_k - \hat{a}\hat{v}_k$
22:    **end for**
23:    $\hat{u}_{\text{prev}} \leftarrow \hat{u}$
24: **end while**
25: **Output:** $y \leftarrow z + Z(\hat{x} - \hat{z})$
```

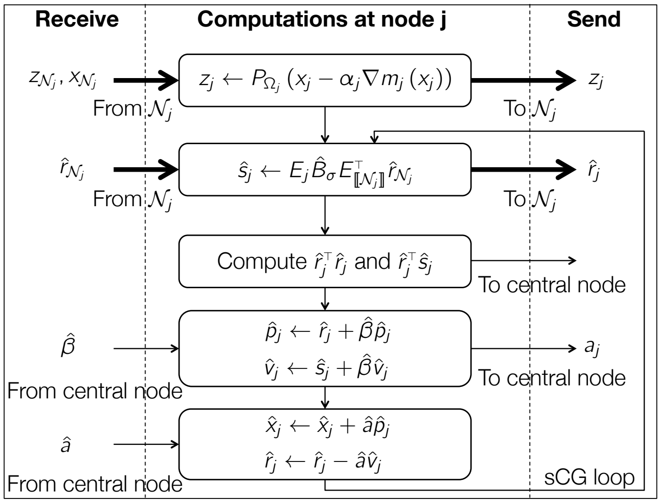
Receive
Computations at node j
Send
$z_{\mathcal{N}_j}, x_{\mathcal{N}_j}$
From $\mathcal{N}_j$
$z_j \leftarrow P_{\Omega_j}(x_j - \alpha_j \nabla m_j(x_j))$
z_j
To $\mathcal{N}_j$
$\hat{r}_{\mathcal{N}_j}$
From $\mathcal{N}_j$
$\hat{s}_j \leftarrow E_j \hat{B}_\sigma E^\top_{[\![\mathcal{N}_j]\!]} \hat{r}_{\mathcal{N}_j}$
$\hat{r}_j$
To $\mathcal{N}_j$
Compute $\hat{r}_j^\top \hat{r}_j$ and $\hat{r}_j^\top \hat{s}_j$
To central node
$\hat{\beta}$
From central node
$\hat{p}_j \leftarrow \hat{r}_j + \hat{\beta}\hat{p}_j$
$\hat{v}_j \leftarrow \hat{s}_j + \hat{\beta}\hat{v}_j$
a_j
To central node
$\hat{a}$
From central node
$\hat{x}_j \leftarrow \hat{x}_j + \hat{a}\hat{p}_j$
$\hat{r}_j \leftarrow \hat{r}_j - \hat{a}\hat{v}_j$
sCG loop

Fig. 1: Workflow at node j in terms of local computations, communications with the set of neighbours $\mathcal{N}_j$ and a central node. Note that we use the index j for a node, and not k, which corresponds to a group of nodes, in which computations are performed in parallel. Thus, the nodes in the set $\mathcal{N}_j$ are not in the same group as node j. Thick arrows represent communications involving vectors, whereas thin arrows stand for communications of scalars. Matrix E_j is defined at Eq. (15).

lines 9 and 16 come with a communication cost, although the amount of transmitted data is very small (one scalar per node). In the end, one should notice that the information that is required to be known globally by all nodes $\{1,\ldots,N\}$ is fairly limited at every iteration of TRAP. It only consists of the trust region radius Δ and the step-sizes $\hat{a}$ and $\hat{\beta}$ in the refinement step 2. Finally, at every iteration, all nodes need to be informed of the success or failure of the iteration so as to update or freeze their local variables. This is the result of the trust region test, which needs to be carried out on a central node. In Figure 1, we give a sketch of the workflow at a generic node j. One can notice that, in terms of local computations, TRAP behaves as a standard two-phase approach on every node.

4 Convergence Analysis

The analysis of TRAP that follows is along the lines of the convergence proof of trust region methods in (Burke et al, 1990), where the Cauchy point is computed via a projected search, which involves a sequence of evaluations of the model function on a central node. However, for TRAP, the fact that the Cauchy point is yielded by an distributed projected gradient step on the model function requires some modifications in the analysis. Namely, the lower bound on the decrease in the model and the upper bound on criticality at the Cauchy point are expressed in a rather different way. However, the arguments behind the global convergence proof are essentially the same as in (Burke et al, 1990).

In this section, for theoretical purposes only, another first-order criticality measure different from (2) is used. We utilise the condition that $x^* \in \Omega$ is a first-order critical point if the projected gradient at x^* is zero,

$$\nabla_\Omega L\left(x^*\right) = 0 \ , \tag{12}$$

where, given $x \in \Omega$, the projected gradient is defined as

$$\nabla_\Omega L\left(x\right) := P_{\mathcal{T}_\Omega(x)}\left(-g\left(x\right)\right) \ .$$

Discussions on this first-order criticality measure can be found in (Conn, A.R. and Gould, N.I.M. and Toint, P.L., 2000). It is equivalent to the standard optimality condition

$$\left\langle g\left(x^*\right), x - x^*\right\rangle \geq 0, \text{ for all } x \in \Omega \ .$$

It follows from Moreau's decomposition that a point x^* satisfying (12) automatically satisfies (2). Consequently, it is perfectly valid to use (2) for the convergence analysis of TRAP.

4.1 Global Convergence to First-order Critical Points

We start with an estimate of the block-coordinate model decrease provided by the Cauchy points z_k, $k \in \{1,\ldots,K\}$, of Algorithm 1. For this purpose, we define for all $k \in \{1,\ldots,K\}$,

$$m_k\left(x'\right) := m\left(z_{[\![1,k-1]\!]}, x', x_{[\![k+1,K]\!]}\right) \ , \tag{13}$$

where $x' \in \mathbb{R}^{n_k}$. This corresponds to the model function evaluated at x' with the block-coordinates 1 to $k-1$ being fixed to the associated Cauchy points $z_1,\ldots,z_{k-1}$ and the block-coordinates $k+1$ to K having values $x_{k+1},\ldots,x_K$. Note that by definition of the function m_k,

$$m_k\left(z_k\right) = m_{k+1}\left(x_{k+1}\right) \ ,$$

for all $k \in \{1,\ldots,K-1\}$.

Lemma 41. *There exists a constant $\chi > 0$ such that, for all $k \in \{1, \ldots, K\}$,*

$$m_k(x_k) - m_k(z_k) \geq \chi \frac{\|z_k - x_k\|_2}{\alpha_k} \min\left\{\varDelta, \frac{1}{1 + \|B(x)\|_2} \frac{\|z_k - x_k\|_2}{\alpha_k}\right\} \quad . \tag{14}$$

Proof. The proof goes along the same lines as the one of Theorem 4.3 in (Moré, J.J., 1988). Yet, some arguments differ, due to the alternating projections. We first assume that condition (5) is satisfied with

$$\alpha_k \geq \nu_4 \quad .$$

Using the basic property of the projection onto a closed convex set, we obtain

$$m_k(x_k) - m_k(z_k) \geq \nu_0 \nu_4 \frac{\|z_k - x_k\|_2^2}{\alpha_k^2} \quad .$$

We then consider the second case when

$$\alpha_k \geq \nu_3 \bar{\alpha}_k \quad .$$

The first possibility is then

$$m_k(z_k(\bar{\alpha}_k)) - m_k(x_k) > \nu_0 \langle \nabla m_k(x_k), z_k(\bar{\alpha}_k) - x_k \rangle \quad .$$

However, by definition of the model function in Eq. (13), the left-hand side term in the above inequality is equal to

$$\begin{aligned} &\langle g_k(x), \bar{z}_k - x_k \rangle + \frac{1}{2} \left\langle \bar{z}_k - x_k, E_k B(x) E_k^\top (\bar{z}_k - x_k) \right\rangle \\ &\qquad\qquad + \left\langle \bar{z}_{[\![1,k-1]\!]} - x_{[\![1,k-1]\!]}, E_{[\![1,k-1]\!]} B(x) E_k^\top (\bar{z}_k - x_k) \right\rangle \\ &= \frac{1}{2} \left\langle \bar{z}_k - x_k, E_k B(x) E_k^\top (\bar{z}_k - x_k) \right\rangle + \langle \nabla m_k(x_k), \bar{z}_k - x_k \rangle \quad , \end{aligned}$$

where, given $k \in \{1, \ldots, K\}$, the matrix $E_k \in \mathbb{R}^{n_k \times n}$ is such that for $i \in \{1, \ldots, n_k\}$,

$$E_k(i, n_1 + \ldots + n_{k-1} + i) = 1 \quad , \tag{15}$$

and all other entries are zero. This yields, by the Cauchy-Schwarz inequality

$$\begin{aligned} \frac{\|B(x)\|_2}{2} \|\bar{z}_k - x_k\|_2^2 &> -(1 - \nu_0) \langle \nabla m_k(x_k), \bar{z}_k - x_k \rangle \\ &\geq \frac{1 - \nu_0}{\bar{\alpha}_k} \|\bar{z}_k - x_k\|_2^2 \end{aligned}$$

Hence,

$$\bar{\alpha}_k \geq \frac{2(1 - \nu_0)}{1 + \|B(x)\|_2} \quad .$$

The second possibility is

$$\|\bar{z}_k - x_k\|_\infty \geq \nu_1 \varDelta \quad .$$

For this case, (Moré, J.J., 1988) provides the lower bound

$$\|z_k - x_k\|_2 \geq \nu_3 \nu_1 \Delta \ .$$

Finally, inequality (14) holds with

$$\chi := \nu_0 \min \left\{ \nu_4, 2\left(1-\nu_0\right), \nu_3\nu_1 \right\} \ .$$

□

From Lemma 41, an estimate of the decrease in the model provided by the Cauchy point z is derived.

Corollary 41 (Sufficient decrease)**.** *The following inequality holds*

$$m\left(x\right) - m\left(z\right) \geq \chi \sum_{k=1}^{K} \frac{\|z_k - x_k\|_2}{\alpha_k} \min \left\{ \Delta, \frac{1}{1+\|B\left(x\right)\|_2} \frac{\|z_k - x_k\|_2}{\alpha_k} \right\} \ . \tag{16}$$

Proof. This is a direct consequence of Lemma 41 above, as

$$m\left(x\right) - m\left(z\right) = \sum_{k=1}^{K} m_k\left(x_k\right) - m_k\left(z_k\right) \ .$$

from the definition of m_k in Eq. (13). □

In a similar manner to (Burke et al, 1990), the level of criticality reached by the Cauchy point z is measured by the norm of the projected gradient of the objective, which can be upper bounded by the difference between the current iterate x and the Cauchy point z.

Lemma 42 (Relative error condition)**.** *The following inequality holds*

$$\|\nabla_\Omega L\left(z\right)\|_2 \leq K\|B\left(x\right)\|_2 \|z-x\|_2 + \sum_{k=1}^{K} \left(\frac{\|z_k - x_k\|_2}{\alpha_k} + \|g_k\left(z\right) - g_k\left(x\right)\|_2 \right) \ . \tag{17}$$

Proof. From the definition of z_k as the projection of

$$x_k - \alpha_k \nabla m_k\left(x_k\right)$$

onto the closed convex set Ω_k, there exists $v_k \in \mathcal{N}_{\Omega_k}\left(z_k\right)$ such that

$$0 = v_k + \nabla m_k\left(x_k\right) + \frac{z_k - x_k}{\alpha_k} \ .$$

Hence,

$$\|v_k + g_k\left(z\right)\|_2 \leq \ \|g_k\left(z\right) - g_k\left(x\right)\|_2 + \|B\left(x\right)\|_2 \|z-x\|_2 + \frac{\|z_k - x_k\|_2}{\alpha_k}$$

However,

$$\left\| P_{\mathcal{N}_{\Omega_k}\left(z_k\right)}\left(-g_k\left(z\right)\right) + g_k\left(z\right) \right\|_2 \leq \|v_k + g_k\left(z\right)\|_2 \ ,$$

and by Moreau's decomposition theorem,

$$-g_k\left(z\right) = P_{\mathcal{N}_{\Omega_k}\left(z_k\right)}\left(-g_k\left(z\right)\right) + P_{\mathcal{T}_{\Omega_k}\left(z_k\right)}\left(-g_k\left(z\right)\right) \quad .$$

Thus,

$$\left\|P_{\mathcal{T}_{\Omega_k}\left(z_k\right)}\left(-g_k\left(z\right)\right)\right\|_2 \leq \left\|g_k\left(z\right) - g_k\left(x\right)\right\|_2 + \left\|B\left(x\right)\right\|_2 \left\|z - x\right\|_2 + \frac{\left\|z_k - x_k\right\|_2}{\alpha_k} \quad .$$

As the sets $\left\{\Omega_k\right\}_{k=1}^{K}$ are closed and convex,

$$\mathcal{T}_{\Omega}\left(z\right) = \mathcal{T}_{\Omega_1}\left(z_1\right) \times \ldots \times \mathcal{T}_{\Omega_K}\left(z_K\right) \quad .$$

Subsequently,

$$\left\|\nabla_{\Omega} L\left(z\right)\right\|_2 \leq \sum_{k=1}^{K} \left\|P_{\mathcal{T}_{\Omega_k}\left(z_k\right)}\left(-g_k\left(z\right)\right)\right\|_2$$

and inequality (17) follows. □

Based on the estimate of the model decrease (16) and the relative error bound (17) at the Cauchy point z, one can follow the standard proof mechanism of trust region methods quite closely (Burke et al, 1990). Most of the steps are proven by contradiction, assuming that criticality is not reached. The nature of the model decrease (16) is well-suited to this type of reasoning. Hence, most of the ideas of (Burke et al, 1990) can be adapted to our setting.

Lemma 43. *If Assumptions 32, 34 and 35 are satisfied, then the sequence of iterates yielded by Algorithm 1 satisfies that for all $k \in \{1, \ldots, K\}$,*

$$\liminf \frac{\left\|z_k - x_k\right\|_2}{\alpha_k} = 0 \quad , \tag{18}$$

Proof. For the sake of contradiction, assume that there exists a block index $k_0 \in \{1, \ldots, K\}$ and $\epsilon > 0$ such that

$$\frac{\left\|z_{k_0}^{l} - x_{k_0}^{l}\right\|_2}{\alpha_{k_0}^{l}} \geq \epsilon$$

for all iteration indices $l \geq 1$. Using Corollary 41, the standard proof mechanism of trust region methods (Burke et al, 1990) can be easily adapted to obtain (18). □

We are now ready to state the main Theorem of this section. It is claimed that all limit points of the sequence $\left\{x^l\right\}$ generated by TRAP are critical points of (1).

Theorem 41 (Limit points are critical points)**.** *Assume that Assumptions 32, 34 and 35 hold. If x^* is a limit point of $\left\{x^l\right\}$, then there exists a subsequence $\left\{l_i\right\}$ such that*

$$\begin{cases} \lim\limits_{i \to +\infty} \left\|\nabla_{\Omega} L\left(z^{l_i}\right)\right\|_2 = 0 \\ z^{l_i} \to x^* \end{cases} \quad . \tag{19}$$

Moreover, $\nabla_{\Omega} L\left(x^*\right) = 0$, meaning that x^* is a critical point of $L + \iota_{\Omega}$.

Proof. Let $\left\{x^{l_i}\right\}$ be a subsequence of $\left\{x^l\right\}$ such that $x^{l_i} \to x^*$. If for all $k \in \{1, \ldots, K\}$

$$\frac{\left\|z_k^{l_i} - x_k^{l_i}\right\|_2}{\alpha_k^{l_i}} \to 0 \ , \tag{20}$$

then the proof is complete, via Lemma 42 and the fact that the step-sizes α_k are upper bounded by ν_5. In order to show (20), given $\epsilon > 0$ one can assume that there exists $k_0 \in \{1, \ldots, K\}$ such that for all $i \geq 1$, $\left\|z_{k_0}^{l_i} - x_{k_0}^{l_i}\right\|_2 / \alpha_{k_0}^{l_i} \geq \epsilon$. One can then easily combine the arguments in the proof of Theorem 5.4 in (Burke et al, 1990) with Corollary 41 and Lemma 42 in order to obtain (19). □

Theorem 41 above proves that all limit points of the sequence $\left\{x^l\right\}$ generated by TRAP are critical points. It does not actually claim convergence of $\left\{x^l\right\}$ to a single critical point. However, such a result can be obtained under standard regularity assumptions (Nocedal, J. and Wright, S., 2006), which ensure that a critical point is an isolated local minimum.

Assumption 41 (Strong second-order optimality condition)**.** *The sequence $\left\{x^l\right\}$ yielded by* TRAP *has a non-degenerate limit point x^* such that for all $v \in \mathcal{N}_\Omega\left(x^*\right)^\perp$, where*

$$\mathcal{N}_\Omega\left(x^*\right)^\perp := \left\{v \in \mathbb{R}^n \ : \ \forall w \in \mathcal{N}_\Omega\left(x^*\right), \ \langle\, w, v\,\rangle = 0\right\} \ , \tag{21}$$

one has

$$\left\langle v, H\left(x^*\right) v\right\rangle \geq \kappa \left\|v\right\|_2^2 \ , \tag{22}$$

where $\kappa > 0$.

Theorem 42 (Convergence to first-order critical points)**.** *If Assumptions 35, 32, 34 and 41 are fulfilled, then the sequence $\left\{x^l\right\}$ generated by* TRAP *converges to a non-degenerate critical point x^* of $L + \iota_\Omega$.*

Proof. This is an immediate consequence of Corollary 6.7 in (Burke et al, 1990). □

4.2 Active-set Identification

In most of the trust region algorithms for constrained optimisation, the Cauchy point acts as a predictor of the set of active constraints at a critical point. Therefore, a desirable feature of the novel Cauchy point computation in TRAP is finite detection of activity, meaning that the active set at the limit point is identified after a finite number of iterations. In this paragraph, we show that TRAP is equivalent to the standard projected search in terms of identifying the active set at the critical point x^* defined in Theorem 42.

Lemma 44. *Given a face $\mathcal{F}$ of Ω, there exists faces $\mathcal{F}_1, \ldots, \mathcal{F}_K$ of $\Omega_1, \ldots, \Omega_K$ respectively, such that $\mathcal{F} = \mathcal{F}_1 \times \ldots \times \mathcal{F}_K$.*

Remark 41. *Given a point $x \in \Omega$, there exists a face $\mathcal{F}$ of Ω such that $x \in \mathrm{ri}\left(\mathcal{F}\right)$. The normal cone to Ω at x is the cone generated by the normal vectors to the active constraints at x. As the set of active constraints is constant on the relative interior of a face, one can write without distinction $\mathcal{N}_\Omega\left(x\right)$ or $\mathcal{N}\left(\mathcal{F}\right)$.*

The following Lemma is similar in nature to Lemma 7.1 in (Burke et al, 1990), yet with an adaptation in order to account for the novel way of computing the Cauchy point. In particular, it is only valid for a sufficiently high iteration count, contrary to Lemma 7.1 of (Burke et al, 1990), which can be written independently of the iteration count. This is essentially due to the fact that the Cauchy point is computed via an alternating projected search, contrary to (Burke et al, 1990), where a centralised projected search is performed.

Lemma 45. *Assume that Assumptions 35, 32, 34 and 41 hold. Let x^* be a non-degenerate critical point of* (1) *that belongs to the relative interior of a face $\mathcal{F}^*$ of Ω. Let $\{\mathcal{F}_k^*\}_{k=1}^K$ be faces of $\{\Omega_k\}_{k=1}^K$ such that $\mathcal{F}^* = \mathcal{F}_1^* \times \ldots \times \mathcal{F}_K^*$ and thus $x_k^* \in \operatorname{ri}(\mathcal{F}_k^*)$, for all $k \in \{1, \ldots, K\}$.*

Assume that $x^l \to x^*$. For l large enough, for all $k \in \{1, \ldots, K\}$ and all $\alpha_k > 0$, there exists $\epsilon_k > 0$ such that

$$x_k^l \in \mathcal{B}\left(x_k^*, \epsilon_k\right) \cap \operatorname{ri}\left(\mathcal{F}_k^*\right) \implies P_{\Omega_k}\left(x_k^l - t_k \nabla m_k\left(x_k^l\right)\right) \in \operatorname{ri}\left(\mathcal{F}_k^*\right) \quad ,$$

for all $t_k \in]0, \alpha_k]$.

Proof. Similarly to the proof of Lemma 7.1 in (Burke et al, 1990), the idea is to show that there exists a neighbourhood of x_k^* such that if x_k^l lies in this neighbourhood, then

$$x_k^l - \alpha_k \nabla m_k\left(x_k^l\right) \in \operatorname{ri}\left(\mathcal{F}_k^* + \mathcal{N}\left(\mathcal{F}_k^*\right)\right) \quad .$$

Lemma 45 then follows by using the properties of the projection operator onto a closed convex set and Theorem 2.3 in (Burke et al, 1990).

For simplicity, we prove the above relation for $k = 2$. It can be trivially extended to all indices k in $\{3, \ldots, K\}$. Let $\alpha_2 > 0$ and $l \geq 1$.

$$x_2^l - \alpha_2 \nabla m_2\left(x_2^l\right) = x_2^l - \alpha_2 g_2\left(x^l\right) - \alpha_2 E_2 B\left(x^l\right) E_1^\top\left(z_1^l - x_1^l\right) \quad ,$$

where the matrix E_k is defined in (15). As x^* is non-degenerate,

$$x^* - \alpha_2 g\left(x^*\right) \in \operatorname{ri}\left(\mathcal{F}^*\right) + \operatorname{ri}\left(\mathcal{N}\left(\mathcal{F}^*\right)\right) \quad .$$

However, as the sets $\{\mathcal{F}_k^*\}_{k=1}^K$ are convex, one has (Rockafellar, R.T. and Wets, R.J.-B., 2009)

$$\operatorname{ri}\left(\mathcal{F}^*\right) = \operatorname{ri}\left(\mathcal{F}_1^*\right) \times \ldots \operatorname{ri}\left(\mathcal{F}_K^*\right) \text{ and } \mathcal{N}\left(\mathcal{F}^*\right) = \mathcal{N}\left(\mathcal{F}_1^*\right) \times \ldots \times \mathcal{N}\left(\mathcal{F}_K^*\right) \quad .$$

Hence,

$$x_2^* - \alpha_2 g_2\left(x^*\right) \in \operatorname{ri}\left(\mathcal{F}_2^*\right) + \operatorname{ri}\left(\mathcal{N}\left(\mathcal{F}_2^*\right)\right) = \operatorname{int}\left(\mathcal{F}_2^* + \mathcal{N}\left(\mathcal{F}_2^*\right)\right) \quad ,$$

by Theorem 2.3 in (Burke et al, 1990). By continuity of the objective gradient g, there exists $\delta_2 > 0$ such that

$$\left\|x^l - x^*\right\|_2 < \delta_2 \implies x_2^l - \alpha_2 g_2\left(x^l\right) \in \operatorname{int}\left(\mathcal{F}_2^* + \mathcal{N}\left(\mathcal{F}_2^*\right)\right) \quad .$$

However, as shown beforehand (Lemma 43),

$$\lim_{l \to +\infty} \left\|z_1^l - x_1^l\right\|_2 = 0 \quad .$$

Moreover, $E_2 B\left(x^l\right) E_1^\top$ is bounded above (Ass. 35), subsequently for l large enough,

$$x_2^l - \alpha_2 \nabla m_2\left(x_2^l\right) \in \operatorname{int}\left(\mathcal{F}_2^* + \mathcal{N}\left(\mathcal{F}_2^*\right)\right) \subseteq \operatorname{ri}\left(\mathcal{F}_2^* + \mathcal{N}\left(\mathcal{F}_2^*\right)\right) \quad ,$$

by Theorem 2.3 in (Burke et al, 1990). Then, Lemma 45 follows by properly choosing the radii ϵ_k so that $\sum_{k=1}^{K} \epsilon_k^2 = \left(\min\left\{\delta_k\right\}_{k=1}^{K}\right)^2$. □

We have just shown that, for a large enough iteration count l, if the primal iterate x^l is sufficiently close to the critical point x^* and on the same face $\mathcal{F}^*$, then the set of active constraints at the Cauchy point z^l is the same as the set of active constraints at x^*.

Theorem 43. *If Assumptions 35, 32, 34 and 41 are fulfilled, then the following holds*

$$\lim_{l \to +\infty} \left\| \nabla_\Omega L\left(x^l\right) \right\|_2 = 0 \quad .$$

Moreover, there exists l_0 such that for all $l \geq l_0$,

$$\mathcal{A}_\Omega\left(x^l\right) = \mathcal{A}_\Omega\left(x^*\right) \quad .$$

Proof. The reasoning of the proof of Theorem 7.2 in (Burke et al, 1990) can be applied using Lemma 45 and line 13 in Algorithm 1. The first step is to show that the Cauchy point z identifies the optimal active set after a finite number of iterations. This is guaranteed by Theorem 2.2 in (Burke et al, 1990), since $\nabla_\Omega L\left(z^l\right) \to 0$ by Theorem 41, and the sequence $\left\{x^l\right\}$ converges to a non-degenerate critical point by Theorem 42. Lemma 45 is used to show that if x^l is close enough to x^*, then the Cauchy point z^l remains in the relative interior of the same face, and thus the active constraints do not change after some point. □

Theorem 43 shows that the optimal active set is identified after a finite number of iterations, which corresponds to the behaviour of the gradient projection in standard trust region methods. This fact is crucial for the local convergence analysis of the sequence $\left\{x^l\right\}$, as fast local convergence rate cannot be obtained if the dynamics of the active constraints does not settle down.

4.3 Local Convergence Rate

In this paragraph, we show that the local convergence rate of the sequence $\left\{x^l\right\}$ generated by TRAP is almost Q-superlinear, in the case where a Newton model is approximately minimised at every trust region iteration, that is

$$B = \nabla^2 L \quad ,$$

in model (3). Similarly to (11), one can define

$$H_\sigma := H + \frac{\sigma}{2} I \quad . \tag{23}$$

To establish fast local convergence, a key step is to prove that the trust region radius is ultimately bounded away from zero. It turns out that the regularisation of the trust region problem (9) plays an important role in this proof. As shown in the next Lemma 46, after a large enough number of iterations, the trust region radius does not interfere with the iterates and an inexact Newton step is always taken at the refinement stage (Line 9 to 13), implying fast local convergence depending on the level of accuracy in the computation of the Newton direction. However, Theorem 7.4 in (Burke et al, 1990) cannot be applied here, since due to the alternating gradient projections, the model decrease at the Cauchy point cannot be expressed in terms of the projected gradient on the active face at the critical point.

Lemma 46. *If Assumptions 35, 32, 34 and 41 are fulfilled, then there exists an index $l_1 \geq 1$ and $\Delta^* > 0$ such that for all $l \geq l_1$, $\Delta^l \geq \Delta^*$.*

Proof. The idea is to show that the ratio ρ converges to one, which implies that all iterations are ultimately successful, and subsequently, by the mechanism of Algorithm 1, the trust region radius is bounded away from zero asymptotically. For all $l \geq 1$,

$$\left|\rho^l - 1\right| = \frac{\left|L\left(y^l\right) - L\left(x^l\right) - \left\langle g\left(x^l\right), y^l - x^l\right\rangle - \frac{1}{2}\left\langle y^l - x^l, H\left(x^l\right)\left(y^l - x^l\right)\right\rangle\right|}{m\left(x^l\right) - m\left(y^l\right)} \quad . \tag{24}$$

However,

$$\begin{aligned} m^l\left(x^l\right) - m^l\left(y^l\right) &= m^l\left(x^l\right) - m^l\left(z^l\right) + m^l\left(z^l\right) - m^l\left(y^l\right) \\ &\geq \frac{\underline{\eta}}{2}\left\|z^l - x^l\right\|_2^2 + \frac{\underline{\sigma}}{2}\left\|y^l - z^l\right\|_2^2 \\ &\geq \frac{\min\left\{\underline{\eta}, \underline{\sigma}\right\}}{2}\left(\left\|z^l - x^l\right\|_2^2 + \left\|y^l - z^l\right\|_2^2\right) \\ &\geq \frac{\min\left\{\underline{\eta}, \underline{\sigma}\right\}}{2}\max\left\{\left\|z^l - x^l\right\|_2^2, \left\|y^l - z^l\right\|_2^2\right\} \quad , \end{aligned}$$

and

$$\begin{aligned} \left\|p^l\right\|_2 &\leq \left\|y^l - z^l\right\|_2 + \left\|z^l - x^l\right\|_2 \\ &\leq 2\max\left\{\left\|y^l - z^l\right\|_2, \left\|z^l - x^l\right\|_2\right\} \quad . \end{aligned}$$

Hence,

$$m^l\left(x^l\right) - m^l\left(y^l\right) \geq \frac{\min\left\{\underline{\eta}, \underline{\sigma}\right\}}{8}\left\|p^l\right\|_2^2 \quad .$$

Moreover, using the mean-value theorem, one obtains that the numerator in (24) is smaller than

$$\frac{1}{2}\psi^l\left\|p^l\right\|_2^2 \quad ,$$

where

$$\psi^l := \sup_{\tau \in [0,1]}\left\|H\left(x^l + \tau p^l\right) - H\left(x^l\right)\right\|_2 \quad . \tag{25}$$

Subsequently, we have

$$\left|\rho^l - 1\right| \leq \frac{4}{\min\left\{\underline{\eta}, \underline{\sigma}\right\}}\psi^l \quad ,$$

and the result follows by showing that p^l converges to zero. Take $l \geq l_0$, where l_0 is as in Theorem 43. Thus, $p^l \in \mathcal{N}\left(\mathcal{F}^*\right)^{\perp}$. However, from the model decrease, one obtains

$$\frac{1}{2}\left\langle p^l, H\left(x^l\right)p^l\right\rangle \leq \left\langle -g\left(x^l\right), p^l\right\rangle \enspace .$$

From Theorem 42, the sequence $\left\{x^l\right\}$ converges to x^*, which satisfies the strong second-order optimality condition 41. Hence, by continuity of the hessian $\nabla^2 L$ and the fact that $\mathcal{A}_{\Omega}\left(x^l\right) = \mathcal{A}_{\Omega}\left(x^*\right)$, one can claim that there exists $l_1 \geq l_0$ such that for all $l \geq l_1$, for all $v \in \mathcal{N}_{\Omega}\left(x^l\right)^{\perp} = \mathcal{N}\left(\mathcal{F}^*\right)^{\perp}$,

$$\left\langle v, H\left(x^l\right)v\right\rangle \geq \kappa \left\|v\right\|_2^2 \enspace .$$

Thus, by Moreau's decomposition, it follows that

$$\begin{aligned}\frac{\kappa}{2}\left\|p^l\right\|_2^2 &\leq \left\langle P_{\mathcal{T}_{\Omega}(x^l)}\left(-g\left(x^l\right)\right) + P_{\mathcal{N}_{\Omega}(x^l)}\left(-g\left(x^l\right)\right), p^l\right\rangle \\ &\leq \left\|P_{\mathcal{T}_{\Omega}(x^l)}\left(-g\left(x^l\right)\right)\right\|_2 \left\|p^l\right\|_2 \enspace ,\end{aligned}$$

since $p^l \in \mathcal{N}\left(\mathcal{F}^*\right)^{\perp}$. Finally, p^l converges to zero, as a consequence of Lemma 42 and the fact that $\left\|z^l - x^l\right\|_2$ converges to 0, by Lemma 42 and the fact that the step-sizes α_k are upper bounded for $k \in \{1,\ldots,K\}$. □

The refinement step in TRAP actually consists of a truncated Newton method, in which the Newton direction is generated by an iterative procedure, namely the distributed sCG described in Algorithm 2. The Newton iterations terminate when the residual $\hat{s}$ is below a tolerance that depends on the norm of the projected gradient at the current iteration. In Algorithm 2, the stopping condition is set so that at every iteration $l \geq 1$, there exists $\xi^l \in \left]0,1\right[$ satisfying

$$\left\|Z^l\left(Z^l\right)^{\top}\left(g_{\sigma^l}\left(x^l\right) + H_{\sigma^l}\left(x^l\right)p^l\right)\right\|_2 \leq \xi^l \left\|Z^l\left(Z^l\right)^{\top} g\left(x^l\right)\right\|_2 \enspace . \tag{26}$$

The local convergence rate of the sequence $\left\{x^l\right\}$ generated by TRAP is controlled by the sequences $\left\{\xi^l\right\}$ and $\left\{\sigma^l\right\}$, as shown in the following Theorem.

Theorem 44 (Local linear convergence). *Assume that the direction p yielded by Algorithm 2 satisfies* (26) *if $\|p\|_{\infty} \leq \gamma^* \Delta$ and $\mathcal{A}_{\Omega}(x) = \mathcal{A}_{\Omega}(x+p)$, given $\gamma^* \in \left]0,\gamma_2\right[$. Under Assumptions 35, 32, 34 and 41, for a small enough $\bar{\sigma}$, the sequence $\left\{x^l\right\}$ generated by* TRAP *converges Q-linearly to x^* if $\xi^* < 1$ is small enough, where*

$$\xi^* := \limsup_{l\rightarrow+\infty} \xi^l \enspace .$$

If $\xi^* = 0$, the Q-linear convergence ratio can be made arbitrarily small by properly choosing $\bar{\sigma}$, resulting in almost Q-superlinear convergence.

Proof. Throughout the proof, we assume that l is large enough so that the active-set is $\mathcal{A}_{\Omega}\left(x^*\right)$ and that p^l satisfies condition (26). This is ensured by Lemma 46 and Theorem 43, as the sequence $\left\{p^l\right\}$ converges to zero. Thus, we can write $Z^l = Z^*$. The orthogonal projection onto the subspace

$\mathcal{N}\left(\mathcal{F}^*\right)^\perp$ is represented by the matrix $Z^*\left(Z^*\right)^\top$. A first-order development yields a positive sequence $\left\{\delta^l\right\}$ converging to zero such that

$$\begin{aligned}
\left\|Z^*\left(Z^*\right)^\top g\left(x^{l+1}\right)\right\|_2 &\leq \left\|Z^*\left(Z^*\right)^\top\left(g\left(x^l\right)+H\left(x^l\right)p^l\right)\right\|_2+\delta^l\left\|p^l\right\|_2 \\
&\leq \frac{2\delta^l}{\kappa}\left\|Z^*\left(Z^*\right)^\top g\left(x^l\right)\right\|_2+\left\|Z^*\left(Z^*\right)^\top\left(g_{\sigma^l}\left(x^l\right)+H_{\sigma^l}\left(x^l\right)p^l\right)\right\|_2 \\
&\qquad +\bar{\sigma}\left\|Z^*\left(Z^*\right)^\top\left(\frac{p^l}{2}+z^l-x^l\right)\right\|_2 \\
&\leq \left(\frac{2\delta^l}{\kappa}+\xi^l\right)\left\|Z^*\left(Z^*\right)^\top g\left(x^l\right)\right\|_2 \\
&\qquad +\bar{\sigma}\left(\frac{1}{\kappa}+\frac{\left\|Z^*\left(Z^*\right)^\top\left(z^l-x^l\right)\right\|_2}{\left\|Z^*\left(Z^*\right)^\top g\left(x^l\right)\right\|_2}\right)\left\|Z^*\left(Z^*\right)^\top g\left(x^l\right)\right\|_2 \quad .
\end{aligned}$$

where the second inequality follows from the last inequality in Lemma 46, and the definition of g_σ in Eq. (11) and H_σ in Eq. (23). However, from the computation of the Cauchy point described in paragraph 3.2 and Assumption 35, the term

$$\frac{\left\|Z^*\left(Z^*\right)^\top\left(z^l-x^l\right)\right\|_2}{\left\|Z^*\left(Z^*\right)^\top g\left(x^l\right)\right\|_2}$$

is bounded by a constant $C>0$. Hence,

$$\frac{\left\|Z^*\left(Z^*\right)^\top g\left(x^{l+1}\right)\right\|_2}{\left\|Z^*\left(Z^*\right)^\top g\left(x^l\right)\right\|_2} \leq \frac{2\delta^l}{\kappa}+\xi^l+\bar{\sigma}\left(\frac{1}{\kappa}+C\right) \quad .$$

Moreover, a first-order development provides us with a constant $\Upsilon>0$ such that

$$\left\|Z^*\left(Z^*\right)^\top g\left(x^l\right)\right\|_2 \leq \left(\hat{B}+\Upsilon\right)\left\|x^l-x^*\right\|_2 \quad .$$

There also exists a positive sequence $\left\{\epsilon^l\right\}$ converging to zero such that

$$\left\|Z^*\left(Z^*\right)^\top g\left(x^{l+1}\right)\right\|_2 \geq \left\|Z^*\left(Z^*\right)^\top H\left(x^*\right)\left(x^{l+1}-x^*\right)\right\|_2-\epsilon^l\left\|x^{l+1}-x^*\right\|_2 \quad .$$

However, since $x^{l+1}-x^*$ lies in $\mathcal{N}\left(x^*\right)^\perp$, $Z^*\left(Z^*\right)^\top\left(x^{l+1}-x^l\right)=x^{l+1}-x^l$. Thus, by Assumption (22),

$$\left\|Z^*\left(Z^*\right)^\top\nabla L\left(x^{l+1}\right)\right\|_2 \geq \left(\kappa-\epsilon^l\right)\left\|x^{l+1}-x^*\right\|_2 \quad ,$$

which implies that, for l large enough, there exists $\bar{\epsilon}\in\,]0,\kappa[$ such that

$$\left\|Z^*\left(Z^*\right)^\top g\left(x^{l+1}\right)\right\|_2 \geq \left(\kappa-\bar{\epsilon}\right)\left\|x^{l+1}-x^*\right\|_2 \quad .$$

Finally,

$$\frac{\left\|x^{l+1}-x^*\right\|_2}{\left\|x^l-x^*\right\|_2} \leq \frac{\hat{B}+\Upsilon}{\kappa-\bar{\epsilon}}\left(\frac{2\delta^l}{\kappa}+\xi^l+\bar{\sigma}\left(\frac{1}{\kappa}+C\right)\right) \quad ,$$

which yields the result. □

5 Numerical Examples

The optimal AC power flow constitutes a challenging class of nonconvex problems for benchmarking optimisation algorithms and software. It has been used very recently in the testing of a novel adaptive augmented Lagrangian technique (Curtis, F.E. and Gould, N.I.M. and Jiang, H. and Robinson, D.P., 2014. To appear in Optimization Methods and Software). The power flow equations form a set of nonlinear coupling constraints over a network. Some distributed optimisation strategies have already been explored for computing OPF solutions, either based on convex relaxations (Lam, A.Y.S. and Zhang, B. and Tse, D.N., 2012) or nonconvex heuristics (Kim, B.H. and Baldick, R., 1997). As the convex relaxation may fail in a significant number of cases (Bukhsh, W.A. and Grothey, A. and McKinnon, K.I.M. and Trodden, P.A., 2013), it is also relevant to explore distributed strategies for solving the OPF in its general nonconvex formulation. Naturally, all that we can hope for with this approach is a local minimum of the OPF problem. Algorithm 1 is tested on the augmented Lagrangian subproblems obtained via a polar coordinates formulation of the OPF equations, as well as rectangular coordinates formulations. Our TRAP algorithm is run as an inner solver inside a standard augmented Lagrangian loop (Bertsekas, D.P., 1982) and in the more sophisticated LANCELOT dual loop (Conn, A. and Gould, N.I.M. and Toint, P.L., 1991). More precisely, if the OPF problem is written in the following form

$$\begin{aligned} \underset{x}{\text{minimise}}\, & f\left(x\right) \\ \text{s.t. } & g\left(x\right) = 0 \\ & x \in \mathcal{X} \quad , \end{aligned} \tag{27}$$

where $\mathcal{X}$ is a bound constraint set, an augmented Lagrangian loop consists in computing an approximate critical point of the auxiliary program

$$\underset{x \in \mathcal{X}}{\text{minimise}}\, L_{\varrho}(x, \mu) := f(x) + \left(\mu + \frac{\varrho}{2} g(x)\right)^{\top} g(x) \tag{28}$$

with μ a dual variable associated to the power flow constraints and $\varrho > 0$ a penalty parameter, which are both updated after a finite sequence of primal iterations in (28). Using the standard first-order dual update formula, only local convergence of the dual sequence can be proven (Bertsekas, D.P., 1982). On the contrary, in the LANCELOT outer loop, the dual variable μ and the penalty parameter ϱ are updated according to the level of satisfaction of the power flow (equality) constraints, resulting in global convergence of the dual sequence (Conn, A. and Gould, N.I.M. and Toint, P.L., 1991). In order to test TRAP, we use it to compute approximate critical points of the subproblems (27), which are of the form (1). The rationale behind choosing LANCELOT instead of a standard augmented Lagrangian method as the outer loop is that LANCELOT interrupts the inner iterations at an early stage, based on a KKT tolerance that is updated at every dual iteration. Hence, it does not allow one to really measure the absolute performance of TRAP, although it is likely more efficient than a standard augmented Lagrangian for computing a solution of the OPF program. Thus, for all cases presented next, we provide the results of the combination of TRAP with a basic augmented Lagrangian and LANCELOT. The augmented Lagrangian loop is utilised to show the performance of TRAP as a bound-constrained solver, whereas LANCELOT is expected to provide better overall performance. All results are compared to the solution yielded by the nonlinear interior-point solver IPOPT (Wächter, A. and Biegler, L.T., 2006) with the sparse linear solver MA27. Finally, it is important to stress that the results presented in this Section are obtained from a preliminary MATLAB implementation, which is designed to handle small-scale problems. The design of a fully distributed software would involve substantial development and testing, and is thus beyond the scope of this paper.

5.1 AC Optimal Power Flow in Polar Coordinates

We consider the AC-OPF problem in polar coordinates

$$
\begin{aligned}
&\text{minimise} \sum_{g \in \mathcal{G}} c_0^g + c_1^g p_g^G + c_2 \left(p_g^G\right)^2 \qquad (29)\\
&\text{s.t.}\\
&\sum_{g \in \mathcal{G}_b} p_g^G = \sum_{d \in \mathcal{D}_b} P_d^D + \sum_{b' \in \mathcal{B}_b} p_{bb'}^L + G_b^B v_b^2\\
&\sum_{g \in \mathcal{G}_b} q_g^G = \sum_{d \in \mathcal{D}_b} Q_d^D + \sum_{b' \in \mathcal{B}_b} q_{bb'}^L - B_b^B v_b^2\\
&p_{bb'}^L = G_{bb} v_b^2 + \left(G_{bb'} \cos\left(\theta_b - \theta_{b'}\right) + B_{bb'} \sin\left(\theta_b - \theta_{b'}\right)\right) v_b v_{b'}\\
&q_{bb'}^L = -B_{bb} v_b^2 + \left(G_{bb'} \sin\left(\theta_b - \theta_{b'}\right) - B_{bb'} \cos\left(\theta_b - \theta_{b'}\right)\right) v_b v_{b'}\\
&\left(p_{bb'}^L\right)^2 + \left(q_{bb'}\right)^2 + s_{bb'} = \left(S_{bb'}^M\right)^2\\
&v_b^L \leq v_b \leq v_b^U\\
&p^L \leq p_g^G \leq p^U\\
&q^L \leq q_g^G \leq q^U\\
&s_{bb'} \geq 0 \ ,
\end{aligned}
$$

which corresponds to the minimisation of the overall generation cost, subject to power balance constraints at every bus b and power flow constraints on every line bb' of the network, where $\mathcal{G}$ denotes the set of generators and $\mathcal{G}_b$ is the set of generating units connected to bus b. The variables p_g^G and q_g^G are the active and reactive power output at generator g. The set of loads connected to bus b is denoted by $\mathcal{D}_b$. The parameters P_d^D and Q_d^D are the demand active and reactive power at load unit d. The letter $\mathcal{B}_b$ represents the set of buses connected to bus b. Variables $p_{bb'}^L$ and $q_{bb'}^L$ are the active and reactive power flow through line bb'. Variables v_b and θ_b denote the voltage magnitude and voltage angle at bus b. Constants v_b^L, v_b^U are lower and upper bounds on the voltage magnitude at bus b. Constants p^L, p^U, q^L and q^U are lower and upper bounds on the active and reactive power generation. It is worth noting that a slack variable $s_{bb'}$ has been added at every line bb' in order to turn the usual inequality constraint on the power flow through line bb' into an equality constraint. The derivation of the optimal power flow problem in polar form can be found in (Zhu, J., 2009).

As a simple numerical test example for TRAP, we consider a particular instance of NLP (29) on the 9-bus transmission network shown in Fig. 2. As in (28), the augmented Lagrangian subproblem is obtained by relaxing the equality constraints associated with buses and lines in (29). The bound constraints, which can be easily dealt with via projection, remain unchanged. One should notice that NLP (29) has partially separable constraints and objective, so that LANCELOT could efficiently deal with it, yet in a purely centralised manner. In some sense, running TRAP in a LANCELOT outer loop can be seen as a first step towards a distributed implementation of LANCELOT for solving the AC-OPF. It is worth noting that the dual updates only require exchange of information between neighbouring nodes and lines. However, each LANCELOT dual update requires a central communication, as the norm of the power flow constraints need to be compared with a running tolerance (Conn, A. and Gould, N.I.M. and Toint, P.L., 1991). For the 9-bus example in Fig. 2, the Cauchy search of TRAP on the augmented Lagrangian subproblem (28) can be carried out in five parallel steps. This can be observed

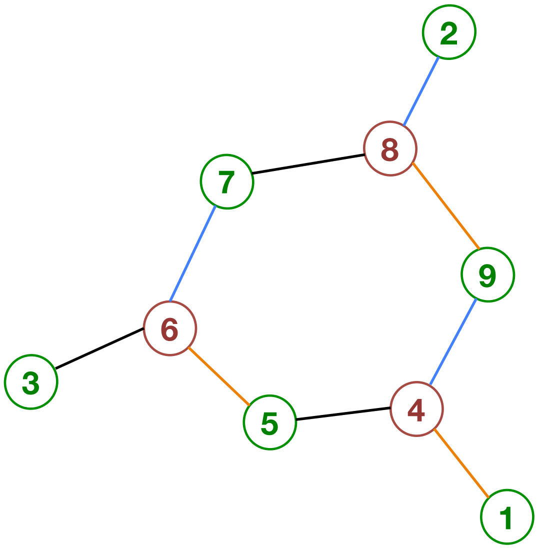

Fig. 2: The 9-bus transmission network from `http://www.maths.ed.ac.uk/optenergy/LocalOpt/`.

by introducing local variables for every bus $b \in \{1, \ldots, 9\}$,

$$x_b := (v_b, \theta_b)^\top \quad ,$$

and for every line

$$bb' \in \Big\{ \{1,4\}, \{4,5\}, \{4,9\}, \{8,9\}, \{2,8\}, \{7,8\}, \{6,7\}, \{3,6\}, \{5,6\} \Big\} \ ,$$

with the line variable $y_{bb'}$ being defined as

$$y_{bb'} := (p_{bb'}, q_{bb'}, s_{bb'})^\top \quad .$$

The line variables $y_{bb'}$ can be first updated in three parallel steps, which corresponds to

$$\left\{y_{\{2,8\}}, y_{\{6,7\}}, y_{\{4,9\}}\right\}, \ \left\{y_{\{7,8\}}, y_{\{3,6\}}, y_{\{4,5\}}\right\}, \ \left\{y_{\{8,9\}}, y_{\{5,6\}}, y_{\{1,4\}}\right\} \quad .$$

Then, the subset

$$\{x_1, x_2, x_3, x_5, x_7, x_9\}$$

can be updated, followed by the subset

$$\{x_4, x_6, x_8\} \quad .$$

As a result, backtracking iterations can be run in parallel at the nodes associated with each line and bus. If a standard trust region Newton method would be applied, the projected search would have to be computed on the same central node without a bound on the number iterations. Thus, the activity detection phase of TRAP allows one to reduce the number of global communications involved in the whole procedure. The results obtained via a basic augmented Lagrangian loop and a LANCELOT outer loop are presented in Tables 1 and 2 below. The data is taken from the archive `http://www.maths.ed.ac.uk/optenergy/LocalOpt/`. In all Tables of this Section, the first column corresponds to the index of the dual iteration, the second column to the number of iterations in the main loop of TRAP at the current outer step, the third column to the total number of sCG iterations at the current outer step, the fourth column to the level of KKT satisfaction obtained at each outer iteration, and the fifth column is the two-norm of the power flow equality constraints at a given dual iteration. To obtain

Outer iter. count	# inner it.	# cum. sCG	Inner KKT	PF eq. constr.
1	79	388	$2.01 \cdot 10^{-7}$	0.530
2	2	40	$2.71 \cdot 10^{-10}$	0.530
3	300	2215	$2.39 \cdot 10^{-2}$	0.292
4	101	2190	$6.50 \cdot 10^{-4}$	$6.56 \cdot 10^{-3}$
5	123	2873	$2.10 \cdot 10^{-3}$	$5.02 \cdot 10^{-6}$
6	56	1194	$4.14 \cdot 10^{-2}$	$1.11 \cdot 10^{-10}$

Table 1: Results for the 9-bus AC-OPF (Fig. 2) using a standard augmented Lagrangian outer loop and TRAP as primal solver. Note that the cumulative number of CG iterations is relatively high, since the refinement stage was not preconditioned.

Outer iter. count	# inner it.	# cum. sCG	Inner KKT	PF eq. constr.
1	37	257	$7.29 \cdot 10^{-2}$	0.530
2	5	25	$1.01 \cdot 10^{-2}$	0.530
3	6	71	$3.23 \cdot 10^{-5}$	0.530
4	100	1330	$8.30 \cdot 10^{-3}$	$4.33 \cdot 10^{-2}$
5	100	1239	$1.80 \cdot 10^{-3}$	$2.53 \cdot 10^{-3}$
6	100	2269	$4.33 \cdot 10^{-2}$	$2.69 \cdot 10^{-5}$
7	64	1541	$3.2 \cdot 10^{-3}$	$1.64 \cdot 10^{-8}$

Table 2: Results for the 9-bus AC-OPF (Fig. 2) using a LANCELOT outer loop and TRAP as primal solver. Note that the cumulative number of CG iterations is relatively high, since no preconditioner was applied in the refinement step.

the results presented in Tables 1 and 2, the regularisation parameter σ in the refinement stage 2 is set to $1 \cdot 10^{-10}$. For Table 1, the maximum number of iterations in the inner loop (TRAP) is fixed to 300 and the stopping tolerance on the level of satisfaction of the KKT conditions to $1 \cdot 10^{-5}$. For Table 2 (LANCELOT), the maximum number of inner iterations is set to 100 for the same stopping tolerance on the KKT conditions. In Algorithm 2, a block-diagonal preconditioner is applied. It is worth noting that the distributed implementation of Algorithm 2 is not affected by such a change. To obtain the results of Table 1, the initial penalty parameter ϱ is set to 10 and is multiplied by 30 at each outer iteration. In the LANCELOT loop, it is multiplied by 100. In the end, an objective value of 2733.55 up to feasibility $1.64 \cdot 10^{-8}$ of the power flow constraints is obtained, whereas the interior-point solver IPOPT, provided with the same primal-dual initial guess, yields an objective value of 2733.5 up to feasibility $2.23 \cdot 10^{-11}$. From Table 1, one can observe that a very tight KKT satisfaction can be obtained with TRAP. From the figures of Tables 1 and 2, one can extrapolate that LANCELOT would perform better in terms of computational time (6732 sCG iterations in total) than a basic augmented Lagrangian outer loop (8900 sCG iterations in total), yet with a worse satisfaction of the power flow constraints ($1.64 \cdot 10^{-8}$ against $1.11 \cdot 10^{-10}$). Finally, one should mention that over a set of hundred random initial guesses, TRAP was able to find a solution satisfying the power flow constraints up to $1 \cdot 10^{-7}$ in all cases, whereas IPOPT failed in approximately half of the test cases, yielding a point of local infeasibility.

5.2 AC Optimal Power Flow on Distribution Networks

Algorithm 1 is then applied to solve two AC-OPF problems in rectangular coordinates on distribution networks. Both 47-bus and 56-bus networks are taken from (Gan, L. and Li, N. and Topcu, U. and Low, S.H., 2014). Our results are compared against the nonlinear interior-point solver IPOPT (Wächter, A. and Biegler, L.T., 2006), which is not amenable to a fully distributed implementation, and the SOCP relaxation proposed by (Gan, L. and Li, N. and Topcu, U. and Low, S.H., 2014), which may be distributed (as convex) but fails in some cases, as shown next. It is worth noting that any distribution network is a tree, so a minimum colouring scheme consists of two colours, resulting in 4 parallel steps for the activity detection in TRAP.

5.2.1 On the 56-bus AC-OPF

On the 56-bus AC-OPF, an objective value of 233.9 is obtained with feasibility $8.00 \cdot 10^{-7}$, whereas the nonlinear solver IPOPT yields an objective value of 233.9 with feasibility $5.19 \cdot 10^{-7}$ for the same initial primal-dual guess.

In order to increase the efficiency of TRAP, following a standard recipe, we build a block-diagonal preconditioner from the hessian of the augmented Lagrangian by extracting block-diagonal elements corresponding to buses and lines. Thus, constructing and using the preconditioner can be done in parallel and does not affect the distributed nature of TRAP. In Fig. 3, the satisfaction of the KKT conditions for the bound constrained problem (28) is plotted for a preconditioned refinement phase and non-preconditioned one. One can conclude from Fig. 3 that preconditioning the refinement phase does not only affect the number of iterations of the sCG Algorithm 2 (Fig. 6), but also the performance of the main loop of TRAP. From a distributed perspective, it is very appealing, for it leads to a strong decrease in the overall number of global communications. Finally, from Fig. 3, it appears that TRAP and a centralised trust region method (with centralised projected search) are equivalent in terms of convergence speed. From Fig. 4, TRAP proves very efficient at identifying the optimal active set in a few iterations (more than 10 constraints enter the active-set in the first four iterations and about 20 constraints are dropped in the following two iterations), which is a proof of concept for the analysis of Section 4. Alternating gradient projections appear to be as efficient as a projected search for identifying an optimal active-set, although the iterates travel on different faces, as shown in Fig. 4. In Fig. 5, the power flow constraints are evaluated after a run of TRAP on program (28). The dual variables and penalty coefficient are updated at each outer iteration. Overall, the coupling of TRAP with the augmented Lagrangian appears to be successful and provides similar performance to the coupling with a centralised trust region algorithm.

Tables 3 and 4 are obtained with an initial penalty coefficient $\rho = 10$ and a multiplicative coefficient of 20.

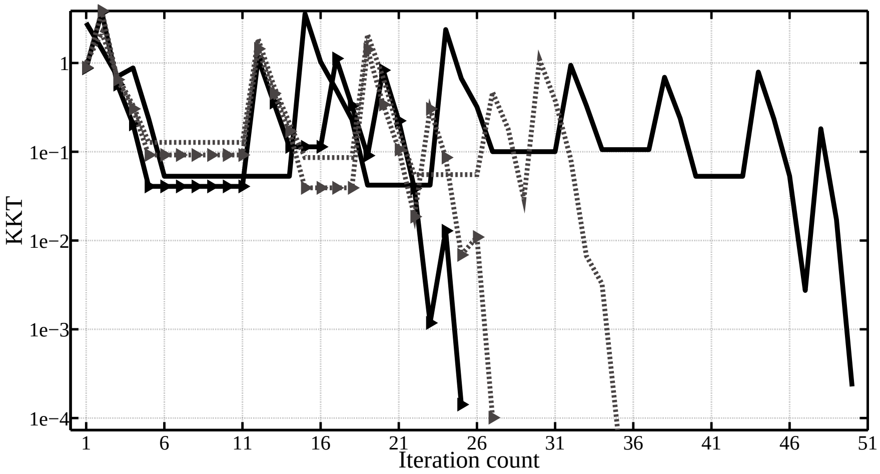

Fig. 3: KKT satisfaction vs iteration count in the fourth LANCELOT subproblem formed on the AC-OPF with 56 buses. When using a centralised projected search as activity detector (dotted grey) and TRAP (full black). Curves obtained with a preconditioned sCG are highlighted with triangle markers.

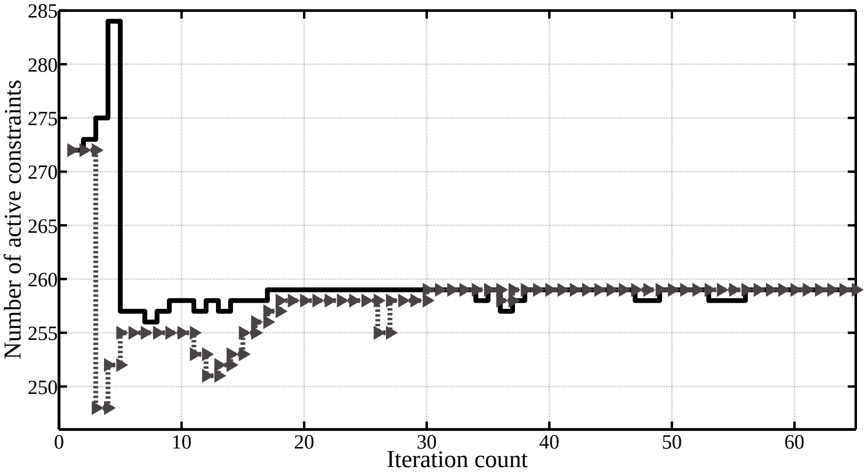

Fig. 4: Active-set history in the first LANCELOT iteration for the 56-bus AC-OPF. Activity detection in TRAP: TRAP (full black), centralised projected search (dashed grey with triangles).

5.2.2 On the 47*-bus AC-OPF*

On the 47-bus AC-OPF, a generating unit was plugged at node 12 (bottom of the tree) and the load at the substation was decreased to 3 pu. On this modified problem, the SOCP relaxation provides a solution, which does not satisfy the nonlinear equality constraints. An objective value of 502.3 is obtained with feasibility $2.57 \cdot 10^{-7}$ for both the AL loop (Tab. 5) and the LANCELOT loop (Tab. 6). The SOCP

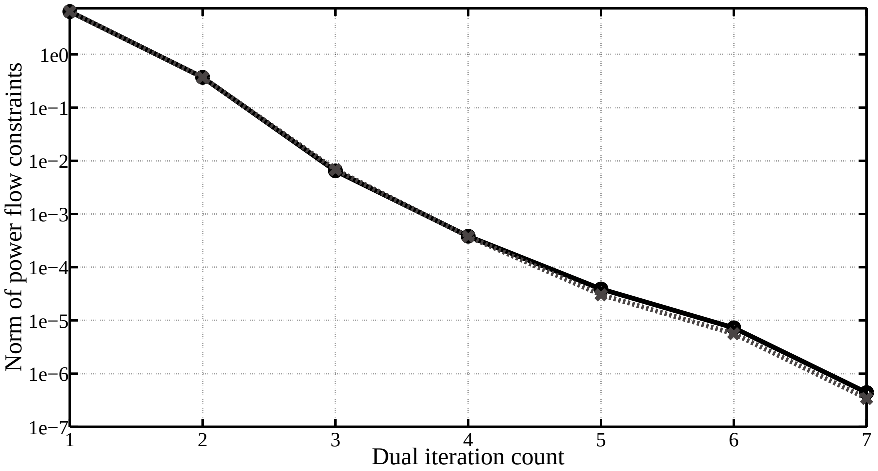

Fig. 5: Norm of power flow constraints on the 56-bus network against dual iterations of a LANCELOT outer loop with TRAP as primal solver. Inner solver: TRAP (full black), centralised trust region method (dashed grey with cross markers).

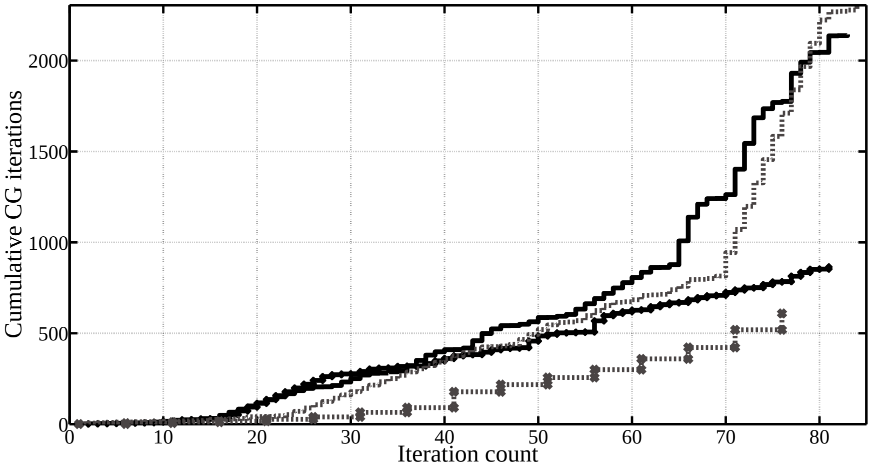

Fig. 6: Cumulative sCG iterations vs iteration count in the first LANCELOT subproblem formed on the AC-OPF with 56 buses. Results obtained with TRAP as inner solver (full black), with a centralised trust region method (dashed grey). Results obtained with a preconditioned refinement stage are highlighted with cross markers.

relaxation returns an objective value of 265.75, but physically impossible, as the power flow constraints are not satisfied. The nonlinear solver IPOPT yields an objective value of 502.3 with feasibility $5.4{\cdot}10^{-8}$.

Outer iter. count	# inner it.	# cum. sCG	Inner KKT	PF eq. constr.
1	122	1382	$8.45 \cdot 10^{-9}$	6.68
2	189	4486	$6.71 \cdot 10^{-9}$	$1.49 \cdot 10^{-1}$
3	139	11865	$9.87 \cdot 10^{-8}$	$8.79 \cdot 10^{-4}$
4	49	3958	$6.75 \cdot 10^{-6}$	$7.92 \cdot 10^{-6}$
5	9	936	$5.45 \cdot 10^{-7}$	$4.58 \cdot 10^{-9}$

Table 3: Results for the 56-bus AC-OPF of (Gan, L. and Li, N. and Topcu, U. and Low, S.H., 2014) using a (local) augmented Lagrangian outer loop with TRAP as primal solver.

Outer iter. count	# inner it.	# cum. sCG	Inner KKT	PF eq. constr.
1	100	924	$9.74 \cdot 10^{-2}$	6.42
2	133	3587	$2.40 \cdot 10^{-3}$	$3.60 \cdot 10^{-1}$
3	54	4531	$1.03 \cdot 10^{-4}$	$4.00 \cdot 10^{-3}$
4	10	858	$4.20 \cdot 10^{-6}$	$1.02 \cdot 10^{-3}$
5	42	3288	$4.37 \cdot 10^{-6}$	$2.32 \cdot 10^{-4}$
6	13	916	$1.82 \cdot 10^{-5}$	$4.35 \cdot 10^{-5}$
7	40	6878	$3.70 \cdot 10^{-7}$	$8.16 \cdot 10^{-6}$
8	6	420	$4.64 \cdot 10^{-6}$	$4.97 \cdot 10^{-7}$

Table 4: Results for the 56-bus AC-OPF of (Gan, L. and Li, N. and Topcu, U. and Low, S.H., 2014) using a LANCELOT outer loop with TRAP as primal solver.

Outer iter. count	# inner it.	# cum. sCG	Inner KKT	PF eq. constr.
1	275	3267	$1.33 \cdot 10^{-7}$	5.80
2	300	7901	$1.39 \cdot 10^{-1}$	$1.12 \cdot 10^{-1}$
3	180	18725	$2.13 \cdot 10^{-6}$	$9.47 \cdot 10^{-5}$
4	26	3765	$5.55 \cdot 10^{-8}$	$6.63 \cdot 10^{-9}$

Table 5: Results for the 47-bus AC-OPF of (Gan, L. and Li, N. and Topcu, U. and Low, S.H., 2014) using an augmented Lagrangian outer loop with TRAP as primal solver.

Outer iter. count	# inner it.	# cum. sCG	Inner KKT	PF eq. constr.
1	180	1147	$8.64 \cdot 10^{-2}$	5.35
2	300	7128	2.23	$3.12 \cdot 10^{-1}$
3	215	11304	$4.65 \cdot 10^{-5}$	$2.97 \cdot 10^{-3}$
4	9	423	$6.05 \cdot 10^{-5}$	$3.28 \cdot 10^{-5}$
5	8	503	$1.11 \cdot 10^{-8}$	$7.90 \cdot 10^{-7}$
6	2	177	$4.64 \cdot 10^{-6}$	$4.03 \cdot 10^{-8}$

Table 6: Results for the 47-bus AC-OPF of (Gan, L. and Li, N. and Topcu, U. and Low, S.H., 2014) using a LANCELOT outer loop with TRAP as primal solver.

6 Conclusions

A novel trust region Newton method, entitled TRAP, which is based on distributed activity detection, has been described and analysed. In particular, as a result of a proximal regularisation of the trust region problem with respect to the Cauchy point yielded by an alternating projected gradient sweep, global and fast local convergence to first-order critical points has been proven under standard regularity assumptions. It has been argued further how the approach can be implemented in distributed platforms. The proposed strategy has been successfully applied to solve various nonconvex OPF problems, for which distributed algorithms are currently raising interest. The performance of the novel activity detection mechanism compares favourably against the standard projected search.

References

Bertsekas, DP (1982) Constrained optimization and Lagrange multiplier methods. Athena Scientific

Bertsekas, DP and Tsitsiklis, JN (1997) Parallel and distributed computation: numerical methods. Athena Scientific

Bolte, J and Sabach, S and Teboulle, M (2014) Proximal alternating linearized minimization for nonconvex and nonsmooth problems. Mathematical Programming 146(1-2):459–494

Bukhsh, WA and Grothey, A and McKinnon, KIM and Trodden, PA (2013) Local solutions of the optimal power flow problem. IEEE Transactions on Power Systems 28(4)

Burke J, Moré J, Toraldo G (1990) Convergence properties of trust region methods for linear and convex constraints. Mathematical Programming 47:305–336

Chiang M, Low S, Calderbank A, Doyle J (2007) Layering as optimization decomposition: A mathematical theory of network architectures. Proceedings of the IEEE 95(1):255–312

Cohen, G (1980) Auxiliary problem principle and decomposition of optimization problems. Journal of Optimization Theory and Applications 32(3):277–305

Conn, A and Gould, NIM and Toint, PL (1991) A globally convergent augmented Lagrangian algorithm for optimization with general constraints and simple bounds. SIAM Journal on Numerical Analysis 28:545–572

Conn, AR and Gould, NIM and Toint, PL (1988) Global convergence of a class of trust region algorithms for optimization with simple bounds. SIAM Journal on Numerical Analysis 25(2)

Conn, AR and Gould, NIM and Toint, PL (2000) Trust Region Methods. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA

Curtis, FE and Gould, NIM and Jiang, H and Robinson, DP (2014. To appear in Optimization Methods and Software) Adaptive augmented Lagrangian methods: algorithms and practical numerical experience. Tech. Rep. 14T-006, COR@L Laboratory, Department of ISE, Lehigh University, URL `http://coral.ie.lehigh.edu/~frankecurtis/wp-content/papers/CurtGoulJianRobi14.pdf`

D'Azevedo, E and Eijkhout, V and Romine, C (1993) LAPACK Working Note 56: Reducing communication costs in the conjugate gradient algorithm on distributed memory multiprocessors. Tech. rep., University of Tennessee, Knoxville, TN, USA

Fei, Y and Guodong, R and Wang, B and Wang, W (2014) Parallel L-BFGS-B algorithm on GPU. Computers and Graphics 40:1–9

Fernández, D and Solodov, MV (2012) Local convergence of exact and inexact augmented Lagrangian methods under the second-order sufficient optimality condition. SIAM Journal on Optimization 22(2):384–407

Gan, L and Li, N and Topcu, U and Low, SH (2014) Exact convex relaxation of optimal power flow in radial network. IEEE Transactions on Automatic Control Accepted for publication

Hamdi, A and Mishra, SK (2011) Decomposition methods based on augmented Lagrangian: a survey. In: Topics in nonconvex optimization, Mishra, S.K.

Hours, J-H and Jones, CN (2014) An augmented Lagrangian coordination-decomposition algorithm for solving distributed non-convex programs. In: Proceedings of the 2014 American Control Conference, pp 4312–4317

Hours, J-H and Jones, CN (2016) A parametric non-convex decomposition algorithm for real-time and distributed NMPC. IEEE Transactions on Automatic Control 61(2), To appear

Kim, BH and Baldick, R (1997) Coarse-grained distributed optimal power flow. IEEE Transactions on Power Systems 12(2)

Lam, AYS and Zhang, B and Tse, DN (2012) Distributed algorithms for optimal power flow. In: Proceedings of the 51st Conference on Decision and Control, pp 430–437

Moré, JJ (1988) Trust regions and projected gradients, Lecture Notes in Control and Information Sciences, vol 113, Springer-Verlag, Berlin

Moreau J (1962) Décomposition orthogonale d'un espace hilbertien selon deux cônes mutuellement polaires. CR Académie des Sciences 255:238–240

Necoara, I and Savorgnan, C and Tran Dinh, Q and Suykens, J and Diehl, M (2009) Distributed nonlinear optimal control using sequential convex programming and smoothing techniques. In: Proceedings of the 48$^{\text{th}}$ Conference on Decision and Control

Nocedal, J and Wright, S (2006) Numerical Optimization. Springer, New-York

Rockafellar, RT and Wets, RJ-B (2009) Variational Analysis. Springer

Steihaug, T (1983) The conjugate gradient method and trust regions in large scale optimization. SIAM Journal on Numerical Analysis 20:626–637

Tran Dinh, Q and Necoara, I and Diehl, M (2013) A dual decomposition algorithm for separable nonconvex optimization using the penalty framework. In: Proceedings of the 52$^{\text{nd}}$ Conference on Decision and Control

Tran-Dinh, Q and Savorgnan, C and Diehl, M (2013) Combining Lagrangian decomposition and excessive gap smoothing technique for solving large-scale separable convex optimization problems. Computational Optimization and Applications 55(1):75–111

Verschoor, M and Jalba, AC (2012) Analysis and performance estimation of the Conjugate Gradient method on multiple GPUs. Parallel Computing 38(10-11):552–575

Wächter, A and Biegler, LT (2006) On the implementation of a primal-dual interior point filter line-search algorithm for large-scale nonlinear programming. Mathematical Programming 106(1):25–57

Xue, D and Sun, W and Qi, L (2014) An alternating structured trust-region algorithm for separable optimization problems with nonconvex constraints. Computational Optimization and Applications 57:365–386

Yamashita, N (2008) Sparse quasi-Newton updates with positive definite matrix completion. Mathematical Programming 115(1):1–30

Zavala, VM and Anitescu, M (2014) Scalable nonlinear programming via exact differentiable penalty functions and trust-region Newton methods. SIAM Journal on Optimization 24(1):528–558

Zavala, VM and Laird, CD and Biegler, LT (2008) Interior-point decomposition approaches for parallel solution of large-scale nonlinear parameter estimation problems. Chemical Engineering Science 63:4834–4845

Zhu, J (2009) Optimization of Power System Operation. IEEE Press, Piscataway, NJ, USA